FORSCHUNGSBERICHT AGRARTECHNIK
des Fachausschusses Forschung und Lehre der
Max-Eyth-Gesellschaft Agrartechnik im VDI (VDI-MEG) **494**

Arne Kuhlmann

Entwicklung, Implementierung und Bewertung eines IT-Systems zur Prozessdokumentation und -unterstützung in der landwirtschaftlichen Nutztierhaltung

Dissertation Hohenheim 2010

Erratum

Forschungsbericht Agrartechnik
des Arbeitskreises Forschung und Lehre der
Max-Eyth-Gesellschaft Agrartechnik im VDI (VDI-MEG)

Arne Kuhlmann

Entwicklung, Implementierung und Bewertung eines IT-Systems zur Prozess-dokumentation und -unterstützung in der landwirtschaftlichen Nutztierhaltung

3 Darstellung und Analyse der Ausgangslage

3.1 Der Versuchsstall und dessen bauliche Gegebenheiten

3.2 Technik im Versuchsstall

3.2.1 Anlagen
3.2.1.1 Fütterungsanlage
3.2.1.2 Lüftungsanlage
3.2.1.3 Waage mit RFID Reader
3.2.1.4 Multigasmonitor
3.2.2 Sensoren und Verbrauchsmesser
3.2.3 Management PC
3.2.4 Handheld

Bitte beachten Sie diese Korrektur auch auf den
Seiten 16 bis 27 und bei Querverweisen im Text.

Universität Hohenheim
Fakultät Agrarwissenschaften
Institut für Agrartechnik
Verfahrenstechnik der Tierhaltungssysteme
Prof. Dr. T. Jungbluth

Entwicklung, Implementierung und Bewertung eines IT-Systems zur Prozessdokumentation und -unterstützung in der landwirtschaftlichen Nutztierhaltung

Dissertation
zur Erlangung des Grades eines Doktors
der Agrarwissenschaften

von
M.Sc. IMIT Arne Kuhlmann
aus Kiel

2010

Die vorliegende Arbeit wurde am 28.04.2010 von der Fakultät Agrarwissenschaften der Universität Hohenheim als "Dissertation zur Erlangung des Grades eines Doktors der Agrarwissenschaften" angenommen.

Prodekan:	Prof. Dr. A. Fangmeier
Hauptberichter:	Prof. Dr. T. Jungbluth
Mitberichter:	Prof. Dr. R. Doluschitz
Mündliche Prüfung:	Prof. Dr. T. Jungbluth
	Prof. Dr. R. Doluschitz
	Prof. Dr. S. Böttinger
Tag der mündlichen Prüfung:	05.08.2010

Die vorliegende Arbeit wurde vom Bundesministerium für Bildung und Forschung (BMBF) im Rahmen des Forschungsverbundprojektes „IT FoodTrace - Konzeption und Entwicklung von IT-Lösungsmodellen zur Steigerung der Nachhaltigkeit durch Qualitätssicherung und Rückverfolgbarkeit in Lieferketten für Lebensmittel tierischer Herkunft“ (Projektkennzeichen 0330761) finanziell gefördert.

Im Selbstverlag:	Arne Kuhlmann
Bezugsquelle:	Universität Hohenheim
	Institut für Agrartechnik – 440 –
	Garbenstr. 9
	70599 Stuttgart

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Fachgebiet Verfahrenstechnik der Tierhaltungssysteme des Instituts für Agrartechnik der Universität Hohenheim.

Mein herzlicher Dank gilt an dieser Stelle Prof. Dr. T. Jungbluth, der mir mit dem entgegengebrachten Vertrauen und den gewährten Freiheiten die erfolgreiche Bearbeitung des Vorhabens ermöglicht hat. Prof. Dr. R. Doluschitz als Mitberichter und Prof. Dr. S. Böttinger danke ich für die engagierte Durchsicht der Arbeit und die konstruktive Kritik.

Dr. D. Herd gilt mein Dank für die immer kompetente Betreuung bei fachlichen Fragen, die in meine Arbeit investierte Zeit und das freundschaftliche Verhältnis.

Am Erfolg des Projektes, insbesondere im technischen Bereich, hat Benjamin Rößler großen Anteil. Ihm danke ich für sein Engagement und, stellvertretend für alle Mitarbeiter und Doktoranden des Instituts, für die vielen fachlichen wie auch nicht fachlichen Gespräche in freundschaftlicher Atmosphäre.

Gedankt sei für ihre Unterstützung auch der Leitung und den Mitarbeitern der Versuchsstation Unterer Lindenhof, den Mitarbeitern der Messtechnik des Instituts für Agrartechnik, allen Projektpartnern sowie den Vertretern kooperierender Firmen.

Schließlich gilt mein Dank all jenen, die mich auf meinem Weg begleitet und motiviert haben.

Der größte Dank gebührt meinen Eltern, da sie mir alle Freiheit gewähren, meine Entscheidungen mittragen und mir immer zur Seite stehen.

Für das große Verständnis für meine Arbeit und meine Launen, die Unterstützung und Motivation danke ich von Herzen Alexandra Zug, der Frau an meiner Seite.

Hohenheim, im September 2010 Arne Kuhlmann

Meinen Eltern

gewidmet

Kurzbeschreibung

In der Nutztierhaltung ist der Einsatz von Automatisierungstechnik verbreitet. Diese erleichtert den Prozessbeteiligten die tägliche Arbeit, indem sie Teilprozessschritte autark abarbeitet. Die Überwachung der eingesetzten Technik erfolgt zumeist manuell. Ebendies gilt für die Erfassung prozessrelevanter Parameter wie Ressourcenverbräuche und Klimadaten. Eine ganzheitliche Prozessüberwachung und –dokumentation ist daher nur mit großem personellen Aufwand zu erreichen.

Die Tierproduktion sieht sich, veranlasst durch den Strukturwandel und den Ruf nach Lebensmittelsicherheit und Rückverfolgbarkeit, vor der Herausforderung, Informationstechnologie einzuführen. Die in diesem Zusammenhang relevanten Themenfelder Erfassung, Haltung, Nutzung und Austausch von Daten auf landwirtschaftlichen Betrieben und in deren Umfeld beleuchtet diese Arbeit am Beispiel der Schweinemast.

Ausgehend von einem Mastschweinestall erfolgt eine Analyse der Gegebenheiten, der Anforderungen und der Umsetzungsmöglichkeiten zur Erreichung der Ziele Prozessdokumentation und -unterstützung. Die infolge dessen abgeleiteten und ergriffenen Maßnahmen haben ein prototypisches System entstehen lassen, das seinen Fokus auf die vollständige Integration der im Stall vorhandenen technischen Komponenten unter Verwendung von Standards legt.

Neben der Vorstellung und der Bewertung des entstandenen Systems wird der konkrete Nutzen für Wissenschaft und Praxis anhand ausgewählter Anwendungsbeispiele dargestellt sowie Optimierungspotentiale hinsichtlich der genutzten Technologien und Standards aufgezeigt.

Abstract

In livestock farming, the use of automation technology is common. Automation technology is able to perform sub-processes, whereby the farmer is supported in his daily work. The data produced by this technology is usually monitored manually. The same applies to the collection of process parameters such as resource consumption and climate data. Therefore overall process monitoring and process documentation require high workload.

Caused by structural change and the demand for food safety and traceability, livestock farming needs to introduce information technology. This document is dealing with the topics collection, storage, usage and exchange of data on farms and in their environment using the example of pig fattening.

A stable for fattening pigs was used to analyse the conditions, requirements and implementation options for achieving the objectives process documentation and process support. Based on the conclusions drawn, a prototype was developed, that focuses on the full integration of all technical components in the stable using communication and data standards.

Besides the presentation and evaluation of the system, concrete benefits for science and practice are presented using selected examples. Furthermore possibilities for improvements regarding the used technologies and standards are pointed out.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

A	Ampere
ADED	Agricultural Data Exchange Dictionary (ISO 11788)
ADIS	Agricultural Data Interchange Syntax (ISO 11787)
API	Application Programming Interface
AuA	Abgabe und Anwendung
AWS	Amazon Web Services
BFL	Bauförderung Landwirtschaft eV
BIRT	Business Intelligence and Reporting Tools
BLE	Bundesanstalt für Landwirtschaft und Ernährung
BMBF	Bundesministerium für Bildung und Forschung
BTU	Bau, Technik und Umwelt in der landwirtschaftlichen Nutztierhaltung
CCTA	Conference on Computer and Computing Technologies in Agriculture
CH_4	Methan
CO_2	Kohlendioxid
CPU	Central Processing Unit
CSV	Comma Separated Values
DB	Datenbank
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
DLG	Deutsche Landwirtschafts-Gesellschaft
DLL	Dynamic Link Library
EFITA	European Federation for Information Technology in Agriculture
FK	Foreign Key
GB	Gigabyte
GIL	Gesellschaft für Informatik in der Landwirtschaft
GUI	Graphical User Interface
h	Stunde
HME	Hohenheimer Messwerterfassung
HTML	Hypertext Markup Language
http(s)	Hypertext Transfer Protocol (Secure)

IDE	Integrated Development Environment
IKT	Informations- und Kommunikationstechnologie
IP	Internet Protocol
ISO	International Organization for Standardization
JAR	Java Archive
Java EE	Java Enterprise Edition
Java SE	Java Standard Edition
JAXB	Java Architecture for XML Binding
JDBC	Java Database Connectivity
JDK	Java Development Kit
JIAC	Joint International Conference
JNI	Java Native Interface
JRE	Java Runtime Environment
JSP	Java Server Pages
JVM	Java Virtual Machine
kg	Kilogramm
KTBL	Kuratorium für Technik und Bauwesen in der Landwirtschaft
l	Liter
LAN	Local Area Network
LKV NRW	Landeskontrollverband Nordrhein-Westfalen
LON	Local Operating Network
MAC-Adresse	Media-Access-Control-Adresse
max.	maximal
MD5	Message-Digest Algorithm 5
min.	minimal
N_2O	Lachgas
Java ME	Java Micro Edition
ml	Milliliter
NH_3	Ammoniak
OS	Operating System (Betriebssystem)
PCI	Peripheral Component Interconnect
PHP	Hypertext Preprocessor
PK	Primary Key (Primärschlüssel)

ppm	parts per million
RAM	Random Access Memory
Repository	Das Verzeichnis des SVN Servers, in dem Projekte liegen
REST	Representational State Transfer
RFID	Radio Frequency Identification
SATA	Serial ATA (Advanced Technology Attachment)
SCHAPI	Schauer Application Programming Interface
s.o.	siehe oben
SQL	Structured Query Language
s.u.	siehe unten
SVN	Subversion
TCP	Transmission Control Protocol
TINI	Tiny Internet Interface
UDP	User Datagram Protocol
UPS	Uninterruptible Power Supply (Unterbrechungsfreie Stromversorgung)
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USB	Universal Serial Bus
usw.	und so weiter
UTF	Unicode Transformation Format
V	Volt
VNC	Virtual Network Computing
Windows CE	Windows Compact Edition
WLAN	Wireless Local Area Network
Xlink	Syntax zur Definition von Links in XML Dokumenten
XLS	Microsoft Excel Dateiformat
XML	Extensible Markup Language
XSD	XML Schema Definition

1 Einleitung

Der Produktionsprozess stellt einen sensiblen Teil in der Wertschöpfungskette tierischer Lebensmittel dar. Da Schweinefleisch den weitaus größten Anteil am Pro-Kopf-Fleischverbrauch der Deutschen ausmacht (BLE, 2008), erfolgte im Rahmen des im Verbundprojekt IT FoodTrace bearbeiteten Teilprojektes „Informations- und Datengewinnung aus Tierhaltungssystemen“[1] die Betrachtung des Schweinemastprozesses. Insbesondere waren hierbei die für die Sicherstellung der Qualitätssicherung und Rückverfolgbarkeit relevanten Prozessdaten von Interesse. Deren Erfassung, Aufbereitung und betriebsinterne Nutzung, aber auch die betriebsexterne Bereitstellung für Prozessverantwortliche und Mitglieder der Wertschöpfungskette sind relevante Aspekte (vgl. VON BORELL et al., 2001).

1.1 Problemstellung

Wirtschaftliche Zwänge, Aufzeichnungspflichten und wachsende Bestandsgrößen legen in der Nutztierhaltung den Einsatz moderner Technologien zum Zwecke der Optimierung des Produktionsprozesses nahe. Entsprechende Softwareprodukte die beim Betriebsmanagement unterstützen sind am Markt verfügbar und werden vorwiegend von Großbetrieben eingesetzt (HOLPP, 2008).
Alle Systeme lassen jedoch eines vermissen: die vollständige Integration der im Stall vorhandenen Anlagen und deren Daten. Eine ganzheitliche Überwachung und eine automatisierte Dokumentation des Produktionsprozesses, im Idealfall auf Einzeltierbasis und unter Einbeziehung der Daten aus anderen Betriebszweigen bzw. von Wirtschaftspartnern, sind bisher nicht möglich (vgl. DOLUSCHITZ et al, 2005).

Elektronische Komponenten, die sich zum Zwecke der Überwachung, Dokumentation und Steuerung von Prozessen auf landwirtschaftlichen Betrieben im Einsatz befinden, bieten nur eingeschränkt Zugriff auf ihre Daten (HERD, D. et al., 2008). Darüber hinaus sind Umfang und Auflösung der zu erfassenden Daten zumeist nicht steuerbar.

[1] Das Forschungsprojekt „Informations- und Datengewinnung aus Tierhaltungssystemen“ ist Teil des BMBF Verbundprojektes „IT FoodTrace“ (Förderkennzeichen: 0330761, Laufzeit Juni 2006 bis Mai 2009), www.itfoodtrace.de

Der Grund für die fehlende Integration liegt in der Schnittstellenproblematik begründet. Ebenso wie im Segment der Landmaschinen Bemühungen unternommen werden, Schnittstellen zu normieren (NOACK, 2007, HENNINGER, 2007 und STEINBERGER et al., 2009), ist dies auch für die Stalltechnik erforderlich.
Der Notwendigkeit, Anlagen miteinander zu vernetzen, steht die Existenz einer Vielzahl herstellerspezifischer Datenschnittstellen gegenüber. Das Vernetzen unterschiedlicher Anlagen sowie das Einbeziehen weiterer Prozessparameter, insbesondere Ressourcenverbräuche und tierindividuelle Merkmale, ist jedoch sinnvoll, um eine ganzheitliche Prozessüberwachung und -bewertung vornehmen zu können.

Ist eine Kommunikationsgrundlage in Form eines Datenaustauschstandards geschaffen und wird dieser von den Herstellern landwirtschaftlicher Anlagen getragen, dürfen Vorteile hinsichtlich des Arbeitszeitbedarfs sowie der Prozessqualität und –effektivität erwartet werden (BERNHARD u. HECKMANN, 2008). Darüber hinaus könnte die Wissenschaft von einer breiten Datenbasis profitieren.

1.2 Zielsetzung

Das Ziel des Teilprojektes „Informations- und Datengewinnung aus Tierhaltungssystemen“ und damit auch Ziel dieser Arbeit, ist die Entwicklung, die Implementierung, der Betrieb und die Bewertung eines prototypischen Systems zur Datenerfassung und –nutzung in der landwirtschaftlichen Nutztierhaltung.
Vorrangiger Zweck dabei ist, einen möglichst hohen Detailgrad der Datenerfassung zu erreichen und neben Qualitäts- und Rückverfolgbarkeitsparametern auch solche zu erheben, die Aussagen über die (Einzel-) Tiergesundheit und die Nachhaltigkeit des Produktionsprozesses zulassen sowie eine Betrachtung der Ressourceneffizienz ermöglichen.
Für die Umsetzung und den Betrieb der Lösung steht der Versuchsstall für Mastschweine auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof der Universität Hohenheim zur Verfügung. Maßgebliche Vorgaben für das Vorhaben sind die folgenden:

- Anbindung aller im Stall vorhandene technischen Geräte, Sensoren und Verbrauchsmesser
- Erfassung prozessrelevanter Daten
- Nutzung geeigneter Standards für den Datenaustausch
- Zentrale Haltung aller erfassten Daten
- Dokumentation des Produktionsprozesses
- Unterstützung des Produktionsprozesses

Aufbauend auf dem anschließenden Testbetrieb des Prototyps konnte dessen Eignung bewertet, Verbesserungspotential identifiziert, Erweiterungsmöglichkeiten beleuchtet und eine Bewertung des Gesamtsystems vorgenommen werden.

Bei der entwickelten Lösung handelt es sich um ein Proof of Concept (zu deutsch: Machbarkeitsstudie), welches sich durch eine beispielhafte, nicht allumfassende und in Teilbereichen unvollständige Umsetzung auszeichnet. Ein Proof of Concept wird typischerweise bei Vorhaben mit zugrundeliegender komplexer Architektur durchgeführt. Häufig erfolgt die Entwicklung eines Prototyps, der eine bessere Einschätzung der Umsetzbarkeit ermöglichen soll. Das Ziel des Proof of Concept besteht nach DERN (2006) darin, „über die Implementierung ausgewählter fachlicher und technischer Anwendungsfälle, die Tragfähigkeit der Architektur […] nachzuweisen." Die Erbringung des Nachweises erfolgt im Falle des Prototyps (nachfolgend Farming Cell genannt) anhand einer Vielzahl implementierter Anwendungsfälle unterschiedlichen Detailgrades.

Ausgehend von dem in Kapitel 2 vorgestellten Stand der Forschung einschließlich geeigneter Standards wird in Kapitel 3 die Ausgangsbasis in baulicher und technologischer Hinsicht dargestellt. Das sich anschließende Kapitel stellt die gewählte Entwicklungsmethode und verwendete Technologien vor. Kapitel 5 geht detailliert auf die für eine Integration der einzelnen Elemente notwendigen Maßnahmen ein, beschreibt deren Durchführung und zeichnet ein ganzheitliches Architekturbild. Kapitel 6 nennt Anwendungsbeispiele, die den Nutzen der Farming Cell verdeutlichen. In Kapitel 7 wird eine Systembewertung vorgenommen und Optimierungspotentiale werden aufgezeigt. Die Arbeit schließt mir einer Zusammenfassung in Kapitel 8.

2 Stand der Forschung

Die Entwicklung von IT Systemen, die verschiedenartige Komponenten integrieren, macht es erforderlich, Daten aus ebendiesen Komponenten zu extrahieren und sie in ein Format zu überführen, das durch die anderen Teilnehmer verarbeitet werden kann. Auch ist die Einigung auf Kommunikationsabläufe erforderlich. Der Projektauftrag, welcher der Farming Cell zugrunde liegt, sieht die Verwendung zweier geeigneter Standards vor, die auf den folgenden Seiten vorgestellt werden.

Verwendete Standards

Die Komplexität moderner IT Systeme erfordert die Verwendung von Standards. Insbesondere die Schnittstellenproblematik, die sich aus der Notwendigkeit der Kommunikation zwischen vernetzten Teilnehmern ergibt, macht den Einsatz von Kommunikationsstandards unabdingbar. Die entwickelte Farming Cell nutzt zwei Standards. Im Bereich der Innenwirtschaft findet der ISO Standard 17532 Stationary equipment for agriculture and forestry - Data communications network for livestock farming, kurz **ISOagriNET**, Anwendung (vgl. Anonymus, 2009a). Für außenwirtschaftliche Zwecke wird die XML basierte Datenaustauschsprache **agroXML** (vgl. KUNISCH et al., 2007) eingesetzt. Beide, ISOagriNET und agroXML, haben gemein, dass sie die Struktur auszutauschender Daten vorgeben. Überdies hinaus, beschreibt ISOagriNET den Kommunikationsablauf und legt das Übertragungsmedium fest, wohingegen agroXML hinsichtlich beider Aspekte freie Hand lässt.

Die Farming Cell ist das erste System, das die beiden noch jungen Standards im Umfeld der Schweinemast implementiert und so Aussagen über deren Eignung in der Praxis ermöglicht.

2.1 ISOagriNET

Der Standard ISOagriNET dient dem Zweck, die Kommunikation zwischen Netzwerkteilnehmern auf dem landwirtschaftlichen Betrieb sowie darüber hinaus zu ermöglichen und zu vereinheitlichen. Er legt seinen Fokus auf zwei Bereiche. Zum einen gibt er mit den Datenformaten Agricultural Data Interchange Syntax (ADIS[2]) oder alternativ XML die **Nachrichtensyntax** vor. Mögliche Inhalte sind im Agricultural

[2] Die ADIS-Syntax ist in dem Internationalen Standard ISO 11787 definiert.

Data Element Dictionary (ADED[3]) hinterlegt. Zum anderen werden **Kommunikationsabläufe** detailliert beschrieben. Den im Standard definierten Kommunikationsabläufen liegen die Prinzipien der im Internet genutzten Netzwerkprotokolle User Datagram Protocol (UDP) und Transmission Control Protocol (TCP) zugrunde. ISOagriNET findet Anwendung bei der Kommunikation zwischen Komponenten innerhalb des Betriebsnetzes und ist darüber hinaus auch für den Datenaustausch über das Internet geeignet (vgl. PAULSEN et al., 2005 und ISO, 2009).

Neben der Verwendung bewährter Netzwerkprotokolle und Datenformate zeichnet ein weiteres Merkmal den Standard ISOagriNET aus: dezentrale Kommunikation. Sind ganzheitliche Managementansätze in Tierhaltungssystemen häufig geprägt von einer zentralen Managementsoftware, welche für das Vorhalten und Verteilen von Informationen zuständig ist, verfolgt ISOagriNET den Ansatz **uni- und bidirektionaler Kommunikation** zwischen zwei Netzwerkteilnehmern. Dieser Ansatz lässt sich auf die logische Netzwerktopologie des vermaschten Netzes übertragen, wie sie folgende Abbildung darstellt.

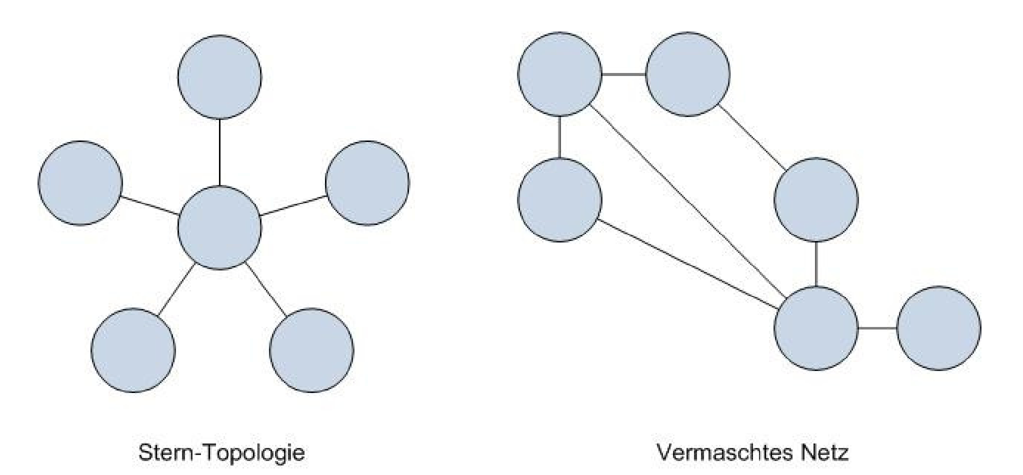


Abbildung 2.1: Netzwerktopologien

Die in Abbildung 2.1 dargestellten Verbindungen zwischen Netzwerkteilnehmern sind logische und bestehen zumeist nur temporär. ISOagriNET, repräsentiert durch die Topologie des vermaschten Netzes, zeichnet sich durch seine Dezentralität aus. Netzwerkteilnehmer können im Falle des vermaschten Netzes direkt-paarweise miteinander kommunizieren. Die Stern-Topologie hingegen erfordert die Einbeziehung des zentralen Teilnehmers in jeden Kommunikationsprozess.

[3] Der Aufbau einer ADED-Datenbank ist in ISO 11788-1 festgelegt.

Neben dem Prinzip der direkt-paarweisen Kommunikation, die auf dem TCP basiert, erlaubt ISOagriNET eine zweite Kommunikationsform: UDP Multicast. Auch diese Art des Datenaustausches ist dezentral angelegt. Netzwerkteilnehmer schließen sich einer virtuellen Gruppe an und tauschen über diese Nachrichten aus. UDP Multicast ist mit der Topologie des vermaschten Netzes vereinbar, denn eine Multicast Gruppe kann als ein Netzwerkteilnehmer verstanden werden, an welchen Nachrichten geschickt werden.

Die beiden genannten Varianten werden im Folgenden näher erläutert und als

- Punkt zu Punkt oder
- Punkt zu Gruppe Kommunikation

bezeichnet.

Die erste Variante basiert auf dem Transmission Control Protocol (TCP). Während des Datenaustausches bauen zwei Kommunikationspartner eine als Session bezeichnete temporäre Verbindung auf. Ist die Kommunikation abgeschlossen, wird die Verbindung wieder abgebaut.

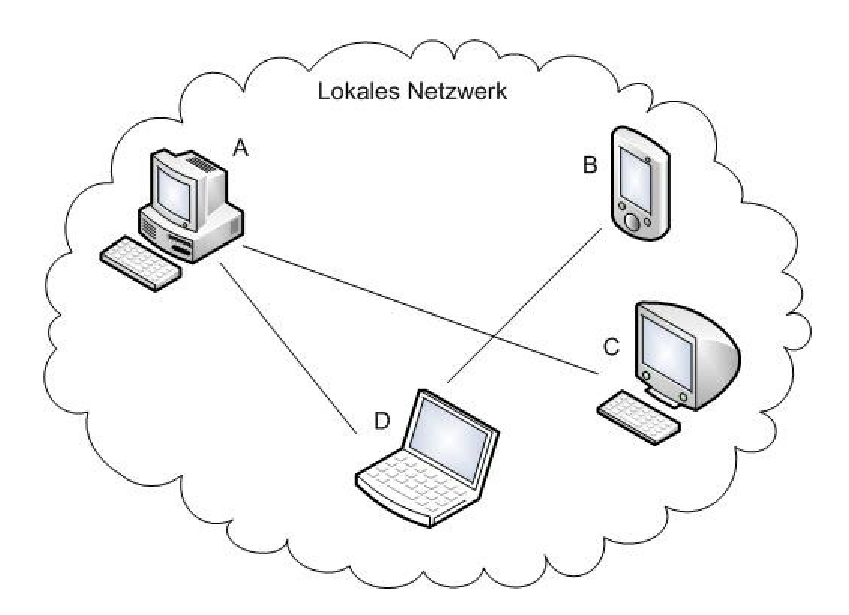
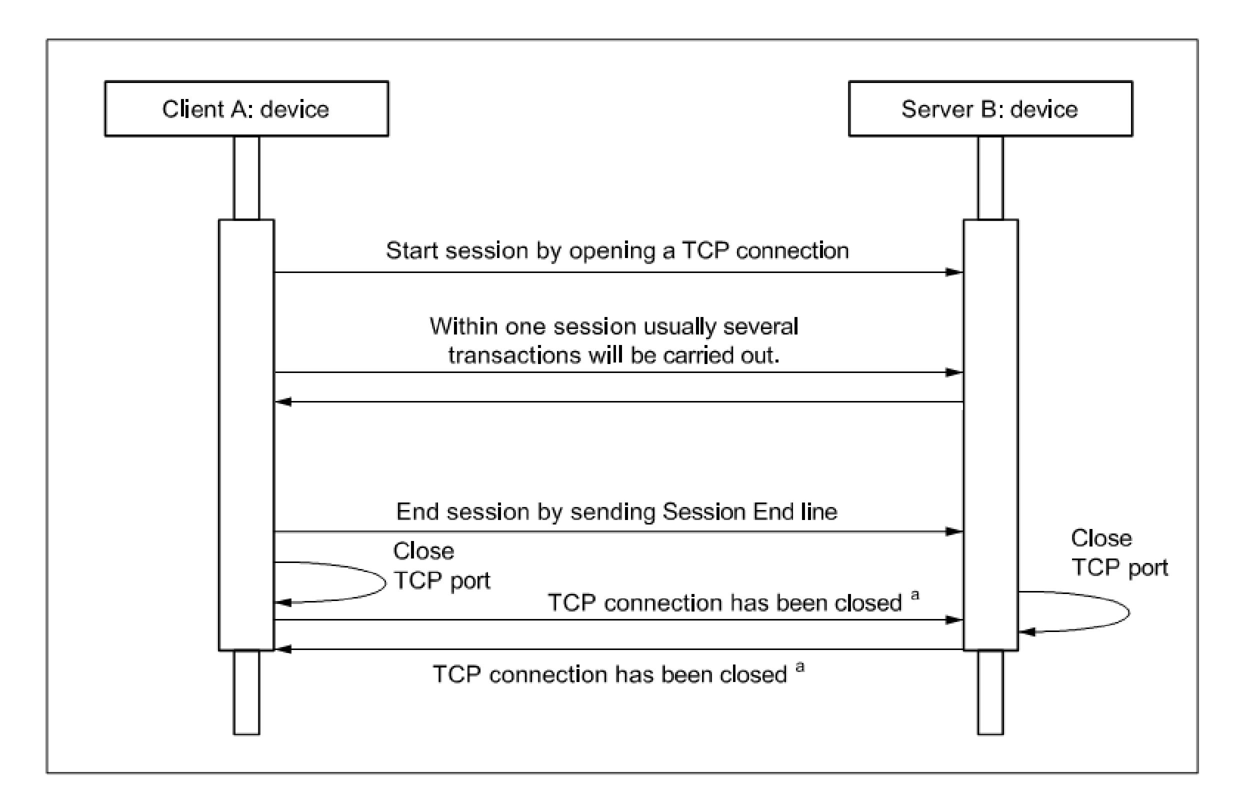


Abbildung 2.2: Punkt zu Punkt Kommunikation

Diesem in Abbildung 2.2 skizzierten Prinzip liegt eine Client-Server-Architektur zugrunde. Der angefragte Kommunikationspartner stellt hierbei die

Serverkomponente dar, wohingegen der Anfragende die Rolle des Clients übernimmt. Das folgende Sequenzdiagramm stellt den Ablauf einer Session dar.



[a] Both these messages are part of the operating system's implementation of the TCP stack.

Abbildung 2.3: Ablauf einer TCP Session nach ISOagriNET
(Abbildung vereinfacht nach ISO, 2007)

Dem gegenüber steht das Prinzip der Punkt-zu-Gruppe Nachrichten, welches einen grundsätzlich anderen Ansatz verfolgt. Es basiert auf dem User Datagram Protocol (UDP). Nachrichten werden nicht direkt zwischen zwei Teilnehmern ausgetauscht, sondern an eine sogenannte Multicast Gruppe gesendet, von wo aus sie an alle Gruppenmitglieder verteilt werden. Eine Multicast Gruppe besitzt eine eindeutige Adresse bestehend aus IP Adresse und Portnummer. Im Fall von ISOagriNET sind dies die IP Adresse 224.111.234.123 und, abhängig vom gewählten Datenformat, die Portnummer 2434 für ADIS oder 2435 für XML.

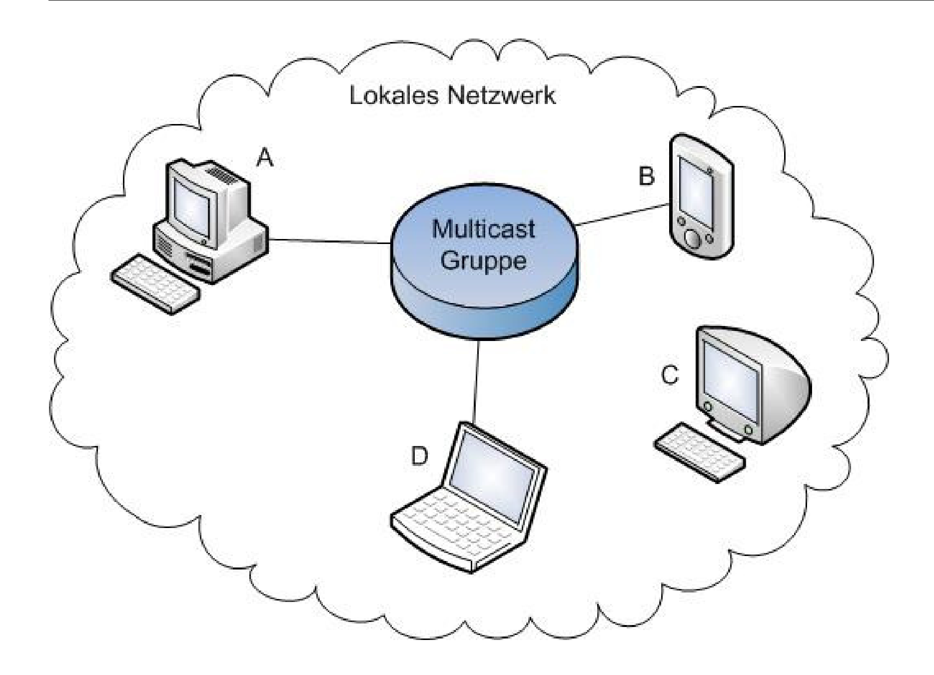


Abbildung 2.4: Punkt zu Gruppe Kommunikation

Abbildung 2.4 zeigt das Prinzip der Multicast Gruppen. Einer Gruppe kann sich jeder im lokalen Netzwerk befindliche Teilnehmer anschließen. Die Netzwerkteilnehmer A, B und D gehören der Multicastgruppe an. Sie können Nachrichten an die Gruppe senden und empfangen alle an diese geschickten Nachrichten. Teilnehmer C hingegen erhält keine Nachrichten der Gruppe und kann auch keine an diese absetzen. Es können mehrere Multicast Gruppen innerhalb eines Netzwerkes existieren, auch kann ein Netzwerkteilnehmer mehreren Gruppen angehören.

Das UDP ist ein verbindungsloses Protokoll, wohingegen das TCP verbindungsorientiert arbeitet. Tabelle 2.1 nennt Vor- und Nachteile der beiden von ISOagriNET vorgesehenen Datenübertragungsvarianten TCP Session und UDP Multicast.

Tabelle 2.1: Vergleich von TCP Session und UDP Multicast

Aspekt	TCP Session	UDP Multicast
Nachrichtenzustellung	sichergestellt	nicht sichergestellt
Netzwerküberlastung	ausgeschlossen	möglich
Identifizierung der Gegenstelle	während Verbindungsaufbau	nach Nachrichtenempfang
Aufwand auf Senderseite für Versand an n Teilnehmer	n mal Vebindungsaufbau, Versand und Verbindungsabbau	einmal Versand
Implementierungsaufwand	mittel	niedrig

Die Wahl der Kommunikationsvariante ist vom individuellen Anforderungsprofil einer Anlage oder Anwendung abhängig. Muss sichergestellt sein, dass eine Nachricht zugestellt wird oder sollte die Notwendigkeit einer unmittelbaren bidirektionalen Kommunikation bestehen, ist die TCP Session von Vorteil. Der Versand unkritischer Nachrichten an einen oder mehrere Teilnehmer kann auf UDP Multicast Basis implementiert werden. Paketverluste treten auch bei UDP in der Praxis selten auf, sofern sich professionelle Hardware, dies gilt insbesondere für Switches, im Einsatz befindet.

Den Aufbau einer Nachricht gibt das **Datenformat** ADIS oder alternativ XML vor. Die konkreten obligatorischen und optionalen **Nachrichtenbestandteile** sind im Data Dictionary (ADED) festgelegt. Da im Rahmen des Projektes Farming Cell das Format XML/ADED aufgrund seines hohen Formatierungsanteils[4] keine Anwendung findet, wird an dieser Stelle lediglich auf das für Massendaten optimierte ADIS/ADED eingegangen.

Das Data Dictionary liefert alle notwendigen Angaben über die Bestandteile der zeilenbasierten **ADIS/ADED** Nachrichten. Den Nutzdatenzeilen werden Zeilen vor- und nachgestellt, welche den Rahmen für die eigentlichen Informationen bilden. Die Struktur einer ADIS/ADED Nachricht ist die folgende:

[4] Eine Nachricht setzt sich zusammen aus der Struktur (Formatierungsanteil) und den eigentlichen Daten (Nutzdatenanteil).

```
Einleitung
Nutzdaten
Schluss
```

Mit Ausnahme der schließenden Zeile, welche lediglich aus der Buchstabenfolge TN (Terminate Normal Data) für das logisches Ende oder ZN (Physical end of file Normal Data) für das physische Ende der Nachricht besteht, setzen sich die Bereiche Einleitung und Nutzdaten jeweils aus einer Definitionszeile und anschließenden Wertezeilen zusammen. Die Einleitung, welche aus einer Definitionszeile (eingeleitet mit DH, Definition Header) und genau einer Wertezeile (eingeleitet mit VH, Value Header) bestehet, wird auch als **Header** bezeichnet. Eine Definitionszeile (eingeleitet mit DN, Definition Normal Data) besitzt auch der **Nutzdatenteil**, die Anzahl der sich anschließenden Wertezeilen (jeweils eingeleitet mit VN, Value Normal Data) hingegen ist beliebig. Eine detaillierte Darstellung der Nachrichten Struktur ist beispielsweise wie folgt möglich.

```
DH...
VH...
DN...
VN...
VN...
...
TN / ZN
```

Den Zeileninhalten liegt eine Struktur aus Objekten (nachfolgend **Entity** genannt) und dazugehörigen Eigenschaften (nachfolgend **Item** genannt) zugrunde. Es ist zulässig, ein Item beliebig vielen Entities zuzuordnen. Ebenso kann eine Entity beliebig viele Items besitzen. Jede Entity, wie auch jedes Item, wird mittels eindeutiger Nummer identifiziert und besitzt einen textuelle Beschreibung sowie weitere Merkmale, wie beispielhaft in Tabelle 2.2 zu sehen.

Zum Zwecke der Erläuterung von Entities und dazugehörigen Items soll die Entity 990054 mit der Bezeichnung **isoagrinet_header** dienen. Diese ist die für ISOagriNET Nachrichten zu verwendende Entity für den Nachrichtenkopf (Header). Die folgende Tabelle ist ein exaktes Abbild der im Data Dictionary der Version 2010 für die Entity 990054 hinterlegten Information.

Tabelle 2.2: Entity 990054 - isoagrinet_header

ID	Typ	Nr	Name	Beschreibung	Daten-typ	Län-ge	Auf-lös-ung	Einheit	Code-set
1	MAN	0	DD-Type	DD-Type	AN	8	0		0
2	MAN	900002	dd_version	ADED DD-Version	N	8	0		0
3	MAN	901059	dd_name	DD_NAME	AN	8	0		0
4	MAN	900003	FIDACREA	Date file create or update	N	8	0	CCYYMMDD	0
5	MAN	900004	FITICREA	File create or update	N	6	0	hhmmss	0
6	OPT	900005	SISSTAT	System Status	AN	1	0		999
7	MAN	900006	SENDER	Sender Name	AN	24	0		999
8	OPT	900007	RECEIVER	Receiver Name	AN	24	0		999
9	OPT	900008	VERSEPRG		AN	8	0		999
10	OPT	900009	dd_national_version	ADED DD national version	AN	8	0		0
11	OPT	901060	dd_national_name	DD_NATIONAL_NAME	AN	8	0		0
12	OPT	900011	PCTYPE	process device type	AN	10	0		0
13	OPT	900012	ADED Manufacturer Version	ADED Manufacturer Version	AN	8	0		0
14	OPT	901054	sender_addr	SENDER_ADDR	AN	99	0	URI	0
15	OPT	901055	reveiver_addr	RECEIVER_ADDR	AN	99	0	URI	0
16	OPT	901056	currency	CURRENCY	AN	3	0		0
17	OPT	901058	character_set	CHARACTER_SET	AN	2	0	AdisCodeset 901058	901058

Jedes Item kann einen Wert eines bestimmten Datentyps, einer definierten Länge und gegebenenfalls Auflösung (Anzahl Nachkommastellen bei numerischen Datentypen) aufnehmen. So lautet die Vorgabe für das Item 900006 SENDER bezüglich dieser Parameter:

- Datentyp: AN (alphanumerisch)
- Länge: 24
- Auflösung: 0

Entsprechend dieser Parameter ist die Senderkennung (SENDER) exakt 24 Zeichen lang anzugeben, Leerzeichen sind gegebenenfalls anzuhängen um diese Länge zu erreichen. Die Korrektheit der Länge ist entscheidend, wie der **Aufbau** von Definitions- und Wertezeile verdeutlicht.

Struktur einer Definitionszeile

```
[Zeilentyp][Entity_Nummer][Item_1_Nummer][Item_1_Länge][Item_1_Auflösung]
. . . [Item_n_Nummer][Item_n_Länge][Item_n_Auflösung]
```

Struktur einer auf die Definitionszeile folgenden Wertezeile

```
[Zeilentyp][Entity_Nummer][Wert_Item_1]. . .[Wert_Item_n]
```

Würde eine in der Definitionszeile vorgegebene Länge eines Items in der Nutzdatenzeile nicht eingehalten, hätte dies eine Linksverschiebung aller folgenden Zeichen zur Folge. Die Zeile wäre in einem solchen Fall nicht mehr eindeutig interpretierbar und müsste vollständig verworfen werden.

Die vorgestellte Struktur soll anhand eines **Beispiels** verdeutlicht werden. Da die bereits genannte Entity isoagrinet_header in allen Nachrichten innerhalb der Farming Cell als Einleitung Verwendung findet, wird diese erläutert. Es sei erwähnt, dass lediglich die obligatorischen Items (Typ = MAN, vgl. Tabelle 2.2) genutzt werden. Dies sind die Items mit den Nummern 0 (DD-Type), 900002 (dd_version), 901059 (dd_name), 900003 (FIDACREA), 900004 (FITICREA) und 900006 (SENDER).

Die beiden Header Zeilen einer Nachricht innerhalb des Farming Cell Netzes könnten beispielsweise wie folgt befüllt sein:

```
DH99005400000000080009000020800090105908000900003080009000040600090000624 0
VH990054AGRO201000002010AGRO201020090721175903HME:00:60:35:07:ee:01
```

Der Aufbau der **Definitionszeile** dieses Headers gestaltet sich wie folgt (Leerzeichen zwischen den Nachrichtenabschnitten sind nachträglich eingefügt):

```
DH990054 00000000080 00900002080 00901059080 00900003080 00900004060
00900006240
```

Tabelle 2.3: Inhalte der Definitionszeile eines ISOagriNET Headers

Nachrichtenabschnitt	Erläuterung
DH990054	Entität 990054 (isoagrinet_header)
00000000 08 0	Item 0 (DD-Type); Länge 8, Auflösung 0
00900002 08 0	Item 900002 (dd_version); Länge 8, Auflösung 0
00901059 08 0	Item 901059 (dd_name); Länge 8, Auflösung 0
00900003 08 0	Item 900003 (FIDACREA); Länge 8, Auflösung 0
00900004 06 0	Item 900004 (FITICREA); Länge 6, Auflösung 0
00900006 24 0	Item 900006 (SENDER); Länge 24, Auflösung 0

Der Aufbau der **Wertezeile** des obigen Headers gestaltet sich wie folgt (Leerzeichen zwischen den Nachrichtenabschnitten sind nachträglich eingefügt):

```
VH990054 AGRO2010 00002010 AGRO2010 20090721 175903 HME:00:60:35:07:ee:01
```

Tabelle 2.4: Inhalte der Wertezeile eines ISOagriNET Headers

Nachrichtenabschnitt	Erläuterung
VH990054	Entität 990054 (isoagrinet_header)
AGRO2010	Wert für Item 0 (DD-Type); AGRO2010
00002010	Wert für Item 900002 (dd_version); 2010
AGRO2010	Wert für Item 901059 (dd_name); AGRO2010
20090721	Wert für Item 900003 (FIDACREA); 21.07.2009
175903	Wert für Item 900004 (FITICREA); 17:59:03
HME:00:60:35:07:ee:01	Wert für Item 900006 (SENDER); HME:00:60:35:07:ee:01

Der Aufbau der sich an die Headerzeilen anschließenden Nutzdatenzeilen entspricht dem vorgestellten Prinzip. Später werden die jeweiligen Nutzdatenzeilen vorgestellt und anhand von Beispielen erläutert.
Das in diesem Kapitel Beschriebene ist nur ein Auszug des laut ISOagriNET Standard Zulässigen. Es ist jedoch hinreichend, um die in den folgenden Kapiteln vorgestellten Nachrichten interpretieren zu können.

2.2 agroXML

Das Datenformat agroXML basiert, wie es sein Name andeutet, auf der Extensible Markup Language (XML). Bei XML handelt es sich um eine sogenannte Auszeichnungssprache (englisch: Markup Language) die dazu dient, Daten in textueller Form hierarchisch darzustellen. Die Festlegung der Ebenen und ihrer Ausprägungen erfolgt durch **XML Schema**, welches eine komplexe Schemabeschreibungssprache darstellt.
Schema Dateien geben detailliert die zulässigen optionalen wie auch obligatorischen Inhalte und deren Struktur vor. Die Beschreibung umfasst neben dem zulässigen Datentyp auch die Kardinalität (Anzahl der Vorkommen) eines Datums. Es sind geschlossene Inhaltslisten (z.B. Geschlecht: m oder w) oder Freitextdaten (z.B. Name) möglich. Ferner findet die XLink Technologie Anwendung, welche die Verknüpfung zweier XML Dokumente mittels eines Uniform Resource Identifier (URI) erlaubt.
Im Gegensatz zu ISOagriNET beschreibt agroXML also lediglich die **Struktur** der Daten, nicht aber, wie ihre Übertragung zu erfolgen hat. Die Art der Weitergabe bleibt dem Nutzer überlassen und kann beispielsweise in Form einer E-Mail, mithilfe eines REST (Representational State Transfer) Services (vgl. Kapitel 5.6.10) oder auch in Papierform erfolgen.

Da agroXML bisher vornehmlich in den Bereichen Pflanzenproduktion und Geodaten Verwendung findet (vgl. STEINBERGER et al., 2007), wurden im Rahmen des Projektes IT FoodTrace durch das KTBL neue Schema Dateien für den Bereich Nutztierhaltung entwickelt (MARTINI et al., 2008). Eine solche Schema Datei ist nachfolgend abgedruckt.

Inhalt der Schema Datei AnimalMeat.xsd:

```
<xsd:schema xmlns="http://www.agroxml.de/schema/devel"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.agroxml.de/schema/devel"
elementFormDefault="qualified" attributeFormDefault="unqualified">
      <xsd:include schemaLocation="CommonBasicComponents.xsd"/>
      <xsd:include schemaLocation="Animal.xsd"/>
      <xsd:element name="Pig" type="PigType">
            <xsd:annotation>
                  <xsd:documentation xml:lang="de">Dieses Element wird in
der Instanz nie verwendet. Anstattdessen werden die konkreten in der
substitutionGroup abgelegten Elemente (Milchkuh, Schwein...)
eingebunden.</xsd:documentation>
            </xsd:annotation>
      </xsd:element>
      <xsd:complexType name="PigType">
            <xsd:annotation>
                  <xsd:documentation>In diesem Datentyp sind alle für die
vorhandenen Tierarten gemeinsamen Elemente
untergebracht.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexContent>
                  <xsd:extension base="AbstractAnimalType">
                        <xsd:sequence>
                              <xsd:element name="EartagNumber"
type="xsd:token" minOccurs="0"/>
                              <xsd:element name="PigRace" type="CodeType"
minOccurs="0"/>
                        </xsd:sequence>
                  </xsd:extension>
            </xsd:complexContent>
      </xsd:complexType>
</xsd:schema>
```

Die Datei AnimalMeat.xsd ist das Modell eines Mastschweins. Eine unter Verwendung dieser Schema Datei erzeugte XML Instanz, d.h. ein konkretes Mastschwein, sieht beispielweise wie folgt aus:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pigType id="_969000000365025" xmlns="http://www.agroxml.de/schema/devel"
xmlns:xlink="http://www.w3.org/1999/xlink">
    <Events>
        <Weighing>
            <DateOrTimestamp>2008-09-12</DateOrTimestamp>
            <Weight uom="kg">128.0</Weight>
        </Weighing>
    </Events>
    <Sex>f</Sex>
    <EartagNumber>969000000365025</EartagNumber>
    <PigRace>PIxDL</PigRace>
</pigType>
```

Die XML Instanz repräsentiert ein weibliches Schwein der Rasse Pietran x Deutsche Landrasse (PIxDL) dessen Ohrmarkennummer 969000000365025 ist. Seine Wiegung am 12.09.2008 ergab ein Gewicht von 128 kg.

3 Darstellung und Analyse der Ausgangslage

In diesem Kapitel wird mit den baulichen Gegebenheiten des Versuchsstalls sowie dessen technischer Ausstattung die **Ausgangslage** vorgestellt. Kapitel 5 schließt an diese Ausführungen an, indem es die zum Erreichen der vollständigen Vernetzung und Datenerfassung erforderlichen Maßnahmen erläutert und zusätzlich benötigte Hard- und Softwarekomponenten vorstellt.

3.1 Der Versuchsstall und dessen bauliche Gegebenheiten

Abbildung 3.1 zeigt den Grundriss des **Versuchsstalls** für Mastschweine auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof, in dem die Farming Cell implementiert ist.
Der Stall ist in zwei Abteile zu jeweils zwei Buchten unterteilt. Jede Bucht bietet 27 Tierplätze á 0,9 m^2 (HÄUSSERMANN, 2006). Ferner existiert ein Vorraum, in dem sich neben Fütterungsanlage und Klimasteuerung auch die Tierwaage mit der RFID Leseeinrichtung befindet. Im Stall sind Sensoren für die Erfassung von Temperatur, Luftfeuchte, Helligkeit, NH_3, CO_2 und Luftdruck, sowie Wärmeenergie-, Wasservolumen- und Strommesser verbaut (vgl. Kapitel 3.2.2). Auf die relevanten Merkmale aller Komponenten wird in den folgenden Abschnitten und detaillierter in Kapitel 5 eingegangen.

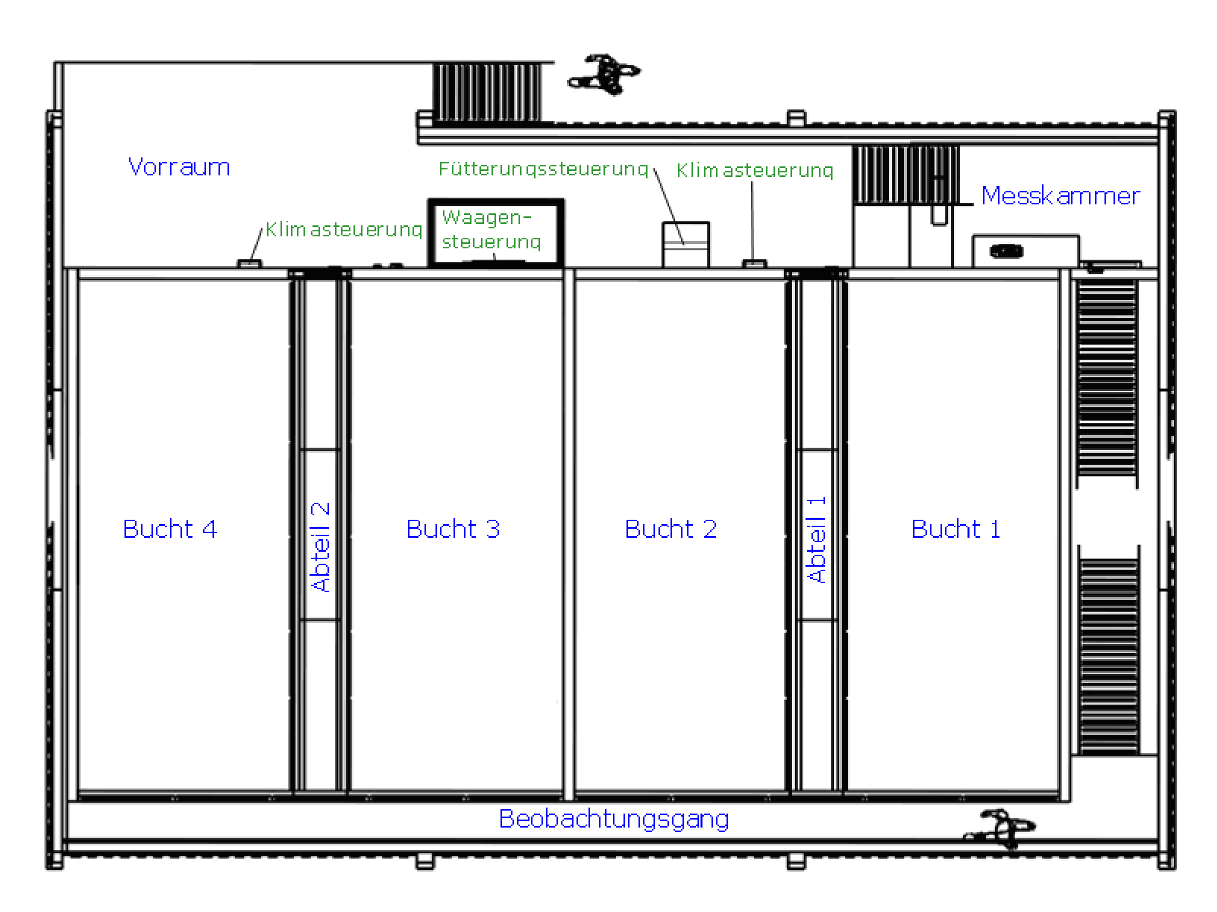


Abbildung 3.1: Grundriss des Versuchsstalls

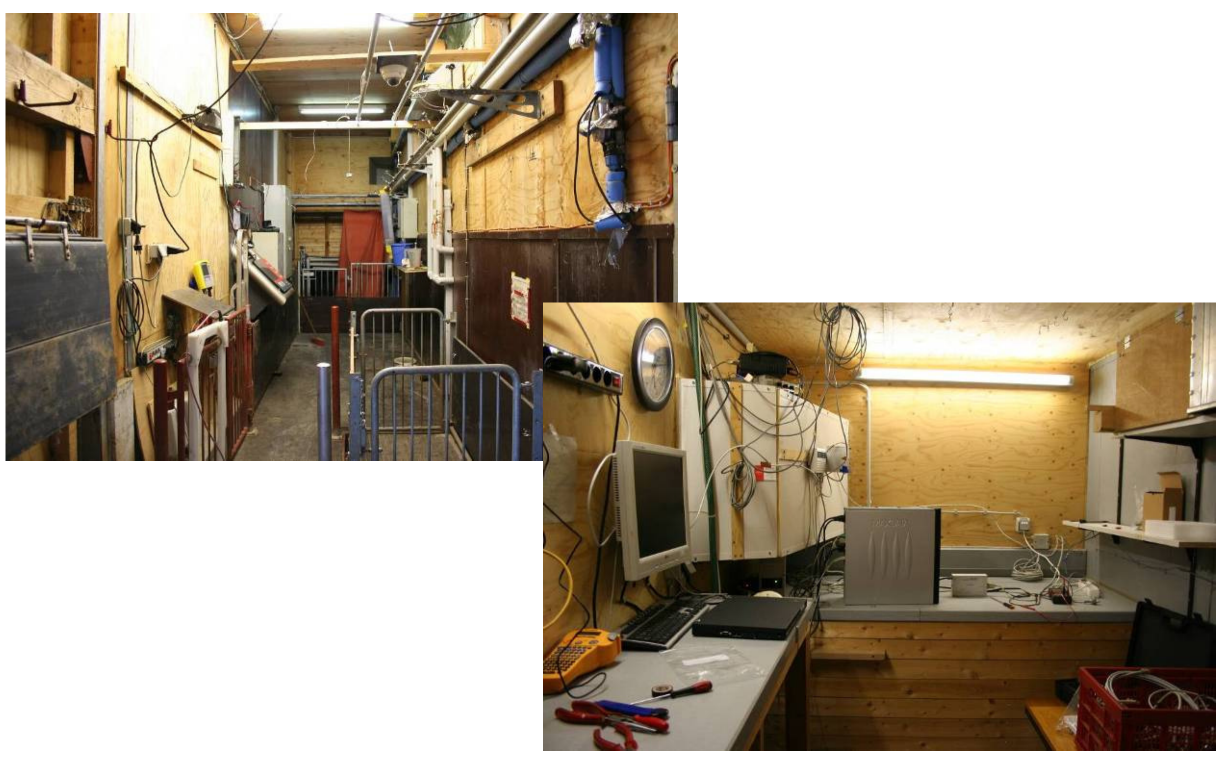


Abbildung 3.2: Vorraum und Messkammer



Abbildung 3.3: Abteil 1 mit Blick zum Beobachtungsgang

Technik im Versuchsstall

Als Ausgangsbasis für das entwickelte System standen ein Computer mit Managementsoftware sowie verschiedene Anlagen, Sensoren und Verbrauchsmesser zur Verfügung oder wurden den Anforderungen entsprechend nachgerüstet. Einige dieser **Komponenten**, insbesondere sind hier die Anlagen Fütterung und Lüftung zu nennen, befinden sich seit mehreren Jahren im Stall. Das Kapitel 3.2 stellt alle Komponenten vor. Dies umfasst neben der Nennung der Eckdaten jeder Komponente das Aufzeigen der Möglichkeiten ihrer Einbindung in das Gesamtsystem unter Verwendung des Standards ISOagriNET. Kapitel 5 greift die Varianten auf und erläutert deren konkrete Umsetzung in Hard- und Software.

3.1.1 Anlagen

Der Versuchsstall ist mit mehreren Anlagen ausgestattet. Bei der verbauten **Fütterungsanlage** handelt es sich um eine Flüssigfütterung der Firma Schauer Maschinenfabrik GmbH und Co KG.

Die **Lüftungsanlage** der Firma Möller Agrarklima GmbH ist in zweifacher Ausführung, separat für beide Abteile, vorhanden.

Die im Vorraum befindliche **Tierwaage** der Firma Tru-Test Ltd ist ausgestattet mit einer Radio Frequency Identification **(RFID)** Leseeinrichtung der Firma Agrident GmbH.
In der Messkammer des Stalls befindet sich ein **Multigasmonitor** der Firma Innova / Lumasense Technologies A/S, der die Schadgaskonzentrationen an verschiedenen Messpunkten innerhalb des Stalls misst und protokolliert.

3.1.1.1 Fütterungsanlage

Bei der verbauten Fütterungsanlage der Firma Schauer Maschinenfabrik GmbH und Co KG handelt es sich um die Sensorflüssigfütterung MEGACOMP - GENPRO 1999. Auf Grund seines Alters verfügt ihr Fütterungscomputer lediglich über eine serielle RS422 (Anonymus, 1998) **Schnittstelle**, die es ermöglicht, die Fütterung von einem Windows PC aus fernzusteuern. Hierfür existiert das Programm Schauer Online. Das für die Kommunikation zwischen PC und Fütterungscomputer eingesetzte Protokoll wurde in Auszügen durch den Hersteller offengelegt.



Abbildung 3.4: Fütterungsanlage im Versuchsstall
Links: Mischbehälter; Rechts: Steuerung

Um eine **Anbindung** des Fütterungscomputers entsprechend dem Standard ISOagriNET zu erreichen, waren drei Schritte durchzuführen.

1.Schnittstellenumsetzung von RS422 auf RJ45 (Ethernet) durch Anschluss an

a) einen Windows PC oder

c) an einen seriellen Server (RS422 auf RJ45).

2. Implementierung einer Software, die die Kommunikation mit dem Fütterungscomputer beherrscht. Dies ist zu erreichen, indem entweder

   a) eine eigenständige Implementierung des Kommunikationsprotokolls vorgenommen wird oder

   b) eine Software Bibliothek durch den Hersteller bereitgestellt wird und diese Anwendung findet.

3. Erweiterung der unter 2. genannten Software um eine ADIS/ADED Schnittstelle zum Zwecke des Datenaustausches mit anderen Netzwerkteilnehmern.

Die entwickelte Lösung stellen die Kapitel 5.3.1 und 5.6.1 vor.

3.1.1.2 Lüftungsanlage

Im Versuchsstall sind zwei Klimacomputer der Firma Möller Agrarklima GmbH verbaut, die jeweils das Klima eines Abteils überwachen und regeln. Das verbaute Modell DR2 bietet über Temperatur- und Feuchtigkeitsfühler hinaus die Möglichkeit chemische Gassensoren anzuschließen.



Abbildung 3.5:Digitaler Klimacomputer DR2

Der Klimacomputer DR2 ist mittels Adapter (LON auf RS232) an die serielle **Schnittstelle** eines Computers anschließbar, welcher ausgestattet mit einem Windows Betriebssystem und der Software Klima 3.37 Hohenheim die Klimacomputer steuern kann. Da das verwendete Kommunikationsprotokoll proprietär ist, war eine **Anbindung** unter ISOagriNET Gesichtspunkten nicht ohne Unterstützung der Firma Möller zu erreichen. Aus diesem Grund und da die Firma Möller einen ISOagriNET fähigen Adapter entwickelt hat, wurde von der Umsetzung

einer eigenen Lösung abgesehen. Kapitel 5.3.2 stellt den Adapter und dessen Funktionsumfang vor.

3.1.1.3 Waage mit RFID Reader

Die **Wiegeeinrichtung** im Versuchsstall besteht aus einem Waagenterminal mit angeschlossener Wiegeplattform (vgl. Abbildung 3.6) und der RFID Leseeinrichtung, welche ebenfalls an das Waagenterminal angeschlossen ist. Die **RFID Leseeinrichtung** setzt sich aus einer Steuer- und Leseeinheit mit angeschlossener Antenne zusammen. Folgende Aufzählung nennt die Modellbezeichnungen der Komponenten.

- Wiegeplattform: Model und Hersteller unbekannt
- Waagenterminal: Model XR3000 der Firma Tru-Test
- RFID Leseeinheit: Model ASR700, Firma Agrident
- RFID Antenne: Model ASA009, Firma Agrident

Parallel zur Möglichkeit, die RFID Steuer- und Leseeinheit an das Waagenterminal anzuschließen, kann diese über eine Schnittstelle (RS232) mit einem Computer (Betriebssystem Windows) verbunden und fernbedient werden. Eine entsprechende Steuerungssoftware stellt die Herstellerfirma bereit. Eine Zuordnung von RFID Nummern zu Tiergewichten ist im Falle des Betriebes der RFID Steuer- und Leseeinheit an einem Computer nicht gegeben, da keine Wiegedaten vom Waagenterminal an die RFID Steuereinheit übergeben werden.

Erfolgt der Anschluss der RFID Steuereinheit hingegen an das Waagenterminal, werden die Nummern erkannter RFID Ohrmarken an dieses weitergereicht. Dort werden aus Tiergewicht und Tierkennung (RFID Nummer) bestehende Datensätze gebildet, welche über die serielle Schnittstelle abrufbar sind (vgl. Anonymus, 2004a). Eine **Integration** des Waagenterminals in das Stallnetz beinhaltet daher gleichsam die der RFID Leseeinheit, da das Waagenterminal als Gateway fungiert.





Abbildung 3.6: Waagenterminal (links) und Waage mit RFID Antenne (rechts)

Um die ISOagriNET konforme Anbindung des Waagenterminals mit angeschlossener RFID Komponente zu erreichen, waren drei Schritte durchzuführen.

1. Umsetzung der Waagenterminalschnittstelle von R232 auf RJ45 (Ethernet) durch Anschluss an
    a) einen Computer oder
    b) eine Hohenheimer Messwerterfassung (HME) oder
    c) einen RS232 auf RJ45 Server.
2. Implementierung einer Software, die das Kommunikationsprotokoll des Waagenterminals umsetzt und eine Abfrage der Wiegedaten (Tierkennung und -gewicht) ermöglicht.
3. Erweiterung der unter 2. genannten Software, so dass eine ISOagriNET konforme Kommunikation mit anderen Netzwerkteilnehmern erfolgen kann.

Die Kapitel 5.3.3 und 5.6.2 stellen die entwickelte Lösung vor.

3.1.1.4 Multigasmonitor

Im Projektverlauf wurde eine Messeinrichtung für Schadgase der Firma Innova / Lumasense Technologies A/S angeschafft. Sie beinhaltet neben dem Multigasmonitor 1412 den Multiplexer 1309 sowie eine Pumpe zum Ansaugen der Probenluft (vgl. Abbildung 3.7).



Abbildung 3.7: Multigasmonitor mit Multiplexer und Pumpe

Der Multigasmonitor verfügt zum Zwecke der Kommunikation mit der Herstellersoftware über eine serielle **Schnittstelle** (RS232). Die Software dient zum einen zur Konfiguration der Messeinrichtung, zum anderen nimmt sie Messwerte entgegen und legt diese in einer Microsoft SQL Server Datenbank ab. Für den Betrieb der Software ist ein Windows Computer erforderlich, das Kommunikationsprotokoll des Multigasmonitors ist nicht offengelegt.
Für die ISOagriNET konforme **Anbindung** der Messeinrichtung waren die folgenden zwei Maßnahmen erforderlich.

1. Schnittstellenumsetzung von RS232 auf RJ45 (Ethernet).
   Verbinden des Multigasmonitors mit einem netzwerkfähigen (RJ45 Schnittstelle) Computer über die RS232 Schnittstelle. Auf dem Computer muss die Hersteller Software mit Microsoft SQL Server Datenbank installiert sein.
2. Implementierung einer Software, welche die Publikation der in der Microsoft SQL Server Datenbank abgelegten Messwerte unter Verwendung der ebendort hinterlegten Konfigurationsparamter des Multigasmonitors (z.B. Messstellenparameter) gemäß ADIS/ADED vornimmt.

Die Nutzung eines Seriellen Servers, der RS232 auf RJ45 umsetzt ist in diesem Fall nicht erforderlich, da sich die Messeinrichtung in unmittelbarer Nähe des mit einer RS232 Schnittstelle ausgestatteten Management PCs (vgl. Kapitel 3.2.3) befindet. Die Umsetzung der genannten Maßnahmen beschreibt das Kapitel 5.3.4.

3.1.2 Sensoren und Verbrauchsmesser

Der Prozess der Schweinemast erfordert zum einen den Einsatz von Ressourcen wie Wasser, Strom und Wärme, zum anderen setzt er Schadgase frei. Um derartige Parameter erfassen zu können, sind Sensoren und Verbrauchsmesser innerhalb des Versuchsstalls und in dessen Außenbereich angebracht. Nachfolgend erfolgt die Vorstellung der verbauten Komponenten, die im Hinblick auf eine automatisierte Datenerfassung von Interesse sind.

Klimadatenerfassung

Die Erfassung klimarelevanter Gase und weiterer Parameter erfolgt mithilfe der in der folgenden Tabelle genannten Sensoren.

Tabelle 3.1: Klimadatenerfassung

Sensor	Hersteller/Anbieter, Typ(en)	Anzahl	Messstelle[5]	Signal
Differenzdruck	Ziehl-Abegg, DSG 200	1	Abteil 1 / Vorraum	Spannung
NH3	ADOS TOX 592	1	Abteil 1	“
CO2	VAISALA, GMP343	1	Abteil 1	“
Temperatur	iButtonLink, DS-2438Z; und andere	3	Abteil 1 und 2, Außenbereich	“
Helligkeit	iButtonLink, unbekannt	1	Abteil 1	“
Feuchtigkeit	iButtonLink, Honeywell HIH-4000	2	Abteil 1, Außenbereich	“

[5] Konfiguration gültig bis 6.11.2009

Verbrauchsdatenerfassung

Die im Einsatz befindlichen Verbrauchsmesser und -zähler sind in Tabelle 3.2 aufgeführt.

Tabelle 3.2: Verbrauchsdatenerfassung

Sensor	Hersteller, Typ(en)	Anzahl	Erfassungsumfang	Signal
Wasservolumen-messer	Zenner, Kalt 10E185; RS Components, Flow Sensor – Dual Range	16	Abteil 1 und 2 gesamt, Fütterung, Befeuchtung, 12 Tränkenippel	Impuls (S0)
Stromenergie-zähler	ShellCount, Drehstromzähler 420465 und Wechselstromzähler EEM12LR	9	Abteil 1 und 2 Beleuchtung, 2 Steckdosen, Befeuchtung, Breifütterung, Flüssigfütterung, Abteil 1 und 2 Belüftung	“
Wärmeenergie-messer	DELTAMESS, TK-WM-ECO	2	Abteil 1 und 2 gesamt	“

Das Ziel der ISOagriNET konformen Integration aller genannten Sensoren und Verbrauchsmesser sowie -zähler in das Gesamtsystem zum Zwecke der regelmäßigen Werterfassung kann nur unter Verwendung zusätzlicher Hardware erreicht werden. Zwei beschrittene Lösungswege zeigen die Kapitel 5.4.1 und 5.4.2 auf.

3.1.3 Management PC

Bei dem in der Messkammer aufgestellten und im Folgenden als Management PC bezeichneten Computer handelt es sich um einen handelsüblichen Desktop PC mit folgender Ausstattung.

- CPU: Intel Core2Duo 6420, 2,13 GHz
- Arbeitsspeicher: 3 GB

- Festplatte: 160 GB SATA
- Externe Schnittstellen: Ethernet, USB, RS232 und weitere
- Betriebssystem: Windows XP Professional

Die Fernsteuerung des Management PCs ist durch den installierten Virtual Network Computing (VNC) Server plattformunabhängig mithilfe eines geeigneten Clients möglich.

Die in der folgenden Tabelle 3.3 genannten **Anwendungen**, die überwiegend zur Steuerung der im Versuchsstall befindlichen Anlagen dienen, sind auf dem Management PC installiert.

Tabelle 3.3: Software auf dem Management PC

Software (Hersteller)	Zweck
Supersau (CLAAS Agrosystems GmbH & Co. KG)	Tierdatenmanagement
Online für Windows (Schauer Maschinenfabrik GmbH)	Steuerung der Fütterungsanlage
Klima 3.3.7 (Möller Agrarklima GmbH)	Steuerung der Klimacomputer
Lumasoft Gas und Microsoft SQL Server 2005 (Innova / Lumasense Technologies A/S)	Steuerung des Multigasmonitors und Messdatenhaltung
Link3000 (Tru-Test)	Steuerung des Waagenterminals
JetPort Commander (Korenix Technology Co., Ltd.)	Verwaltung der seriellen Server
VNC Server (RealVNC Limited)	Software für Remote Zugriff auf den Management PC

Die Einbindung des Management PCs in das Stallnetz ist auf Hardwareebene durch seine Ethernetschnittstelle ISOagriNET konform möglich. Die installierten Anwendungen unterstützen den Standard hingegen nicht. Deren Anpassung bzw. Substitution erläutert Kapitel 5.6.

3.1.4 Handheld

Als mobil nutzbares System ist der Handheld WORKABOUT PRO C der Firma Psion Teklogix Inc. mit integriertem **RFID** Lesemodul AIR200[6] der Firma Agrident GmbH vorhanden (vgl. Abbildung 3.8). Das Gerät ist in der Lage, die der Norm FDX-B (vgl. ISO, 2009) entsprechenden elektronischen Ohrmarken der Masttiere zu lesen. Auch besitzt er ein **WLAN** Modul und ist somit netzwerkfähig. Das installierte Betriebssystem ist Windows CE.

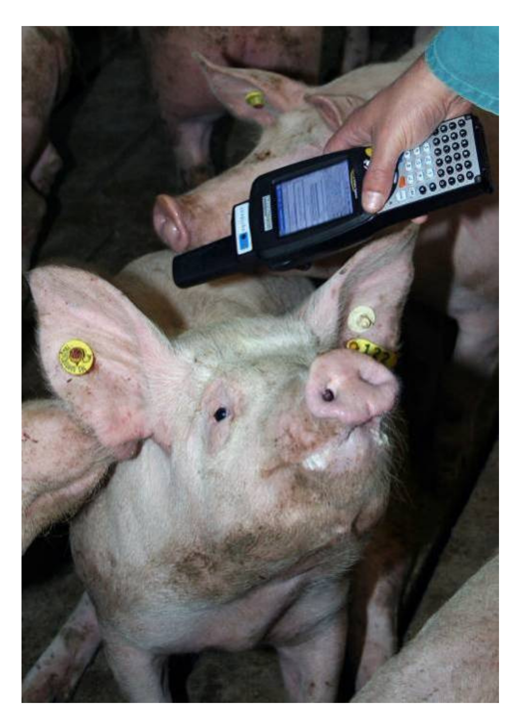


Abbildung 3.8: Handheld mit RFID Lesemodul

Prozessunterstützende Software für den Handheld existierte zu Projektbeginn nicht. Diese wurde prototypisch implementiert und wird in Kapitel 5.6.8 vorgestellt.

[6] Handbuch verfügbar unter http://www.warok.de/tl_files/images/RFID_Web/Agrident_AIR200_GB.pdf

4 Methode

Ein Entwicklungsprojekt wie es Farming Cell darstellt, macht es erforderlich, im Vorfeld Festlegungen hinsichtlich der **Prozesse** und **Werkzeuge** zu treffen, die das Vorhaben zum Erfolg führen sollen. Ist die eigentliche Entwicklung abgeschlossen, bildet die Bewertung der erarbeiteten Lösung den Abschluss des Projektes. Die gewählte Entwicklungsmethode und die für die Umsetzung verwendeten Technologien werden auf den folgenden Seiten vorgestellt.

Die Auswahl der Entwicklungsmethode, der verwendeten Technologien und der Bewertungsparameter wurde durch die im November 2007 bekannten und im Folgenden genannten **Rahmenbedingungen** des Projektes Farming Cell beeinflusst:

- Aufbau, Betrieb und Bewertung eines prototypischen Systems (Farming Cell) als Ziel.
- Zeitrahmen von 18 Monaten für Entwurf, Implementierung und Test des Systems.
- Wahlfreiheit bei technologischen Entscheidungen (Programmiersprachen, Datenbanken etc.).
- Notwendigkeit der Kooperation mit diversen Projektpartnern und Dritten.
- Entwicklungsleistung nahezu vollständig durch Projektmitarbeiter zu erbringen.
- Vergleichbare Systeme nicht existent, daher intensiver Austausch zwischen Entwickler und Kunde sowie umfangreiches Testen notwendig.

4.1 Entwicklungsmethode

Die Wahl geeigneter Methoden für den Entwicklungs- und den Implementierungsprozess ist entscheidend für den Erfolg jedes Softwareentwicklungsprojektes. Neben althergebrachten Ansätzen (vgl. POMBERGER u. PREE, 2004) existieren seit dem letzten Jahrzehnt grundlegend neue. Der Unterschied zwischen alten und neuen Ansätzen soll an dieser Stelle herausgearbeitet und abschließend die gewählten Methoden vorgestellt werden.

Klassische, auch schwergewichtig genannte, Entwicklungsmethoden zeichnen sich durch

- Komplexe und starre Vorgehensmodelle,
- umfassende und detaillierte Planung zu Projektbeginn und damit einhergehende Inflexibilität,
- umfangreiche Dokumentationspflichten sowie
- späte Softwaretests

aus (vgl. GÖTZENAUER, 2006).

Probleme schwergewichtiger Entwicklungsmethoden:
Eine agile[7] Anpassungsfähigkeit an die jeweiligen projektindividuellen Rahmenbedingungen und im Projektverlauf auftretende unvorhergesehene Ereignisse ist nur eingeschränkt gegeben (vgl. GÖTZENAUER, 2006).

Im Verlauf der 90er Jahre entstanden daher sogenannte **leichtgewichtige Entwicklungsmethoden**, deren Ansätze der beschriebenen Problematik Rechnung tragen. Ihnen gemein sind die vier im Jahre 2001 im Agilen Manifest festgehaltenen Paradigmen.

Die vier Paradigmen des Agilen Manifestes (interpretiert nach BECK et al., 2001):

1. Zwischenmenschliche Kommunikation und Interaktion werden höher geschätzt als das Einhalten von Prozessen und die Verwendung bestimmter Werkzeuge.
2. Lauffähige und kundendienliche Software wird höher geschätzt, als die Dokumentation des Entwicklungsprozesses.
3. Die Erarbeitung einer Lösung in Zusammenarbeit mit dem Kunden steht vor Vertragsinhalten.
4. Flexibel auf neue Situationen zu reagieren ist wichtiger, als das Verfolgen eines Planes.

Zu der so entstandenen Gruppe agiler Vorgehensweisen gehört die für den Aufbau der Farming Cell verwendete namens **Crystal Clear**[8]. Diese auf möglichst

[7] Agil: von lateinisch agilis: leichtbeweglich, schnell, rasch, gewandt
[8] Crystal clear: englisch für glasklar, kristallklar

vollständige Transparenz setzende Vorgehensweise lässt sich anhand folgender Prinzipien charakterisieren (vgl. COCKBURN, 2005):

- Räumliche Nähe der Projektbeteiligten.
- Formlose und zeitnahe Kommunikation mit Kunden und Partnern.
- Offenheit in der Kommunikation ohne Repressalien befürchten zu müssen; Kritik und Verbesserungsvorschläge werden laufend geäußert.
- Fokussierung auf ein Projekt.
- Erwartungsstau, erhöhten Erklärungsbedarf, Fehlentwicklungen und Missverständnisse durch häufige Veröffentlichungen und Tests von Zwischenversionen vermeiden.
- Verwendung einer Versionsverwaltung.

Als Implementierungsmethode fand **Experimentelles Prototyping** Anwendung. Dies bedeutet, dass verschiedene Realisierungsansätze verfolgt und unter Umständen wieder verworfen werden. Ziel ist es, geeignete Wege der Realisierung zu identifizieren und Erfahrungswerte zu liefern, mit deren Kenntnis in Folgeprojekten vollwertige Produkte entstehen können (vgl. WALLMÜLLER, 2001). Dieses Ziel deckt sich mit dem in Kapitel 1.2 vorgestellten Anspruch, ein Proof of Concept durchzuführen. Die wesentlichen verwendeten Technologien und Werkzeuge stellt das folgende Kapitel 4.2 vor.

4.2 Verwendete Technologien und Werkzeuge

In diesem Kapitel wird auf die wichtigsten im Rahmen des Softwareentwicklungsprozesses verwendeten und die für den Betrieb der Farming Cell erforderlichen Technologien und Werkzeuge eingegangen.

Java

Alle in dieser Ausarbeitung beschriebenen selbstentwickelten Applikationen wurden mittels der Java Technologie implementiert. Als Plattformen dienten die Java Standard Edition (Java SE), die Java Enterprise Edition (Java EE) sowie eine Version für Mikrocontroller der Firma Maxim Integrated Products (vgl. Kapitel 5.4.1 und 5.6.4). Über Java soll im Folgenden ein kurzer Überblick gegeben werden.

Teil der Java-Technologie der Firma Sun Microsystems sind neben der Programmiersprache Java die Java-Plattformen. Diese stellt Sun Microsystems in drei verschiedenen Ausführungen zur Verfügung (vgl. Tabelle 4.1 und NOURIE u. PAWALAN, 2007).

Tabelle 4.1: Java Plattformversionen der Firma Sun Microsystems

Java Plattformversion	Vornehmlicher Einsatzbereich
Standard Edition (SE)	Desktop Applikationen
Enterprise Edition (EE)	große webbasierte Applikationen
Micro Edition (ME)	Applikationen für mobile Geräte

Im Rahmen des Projektes fanden die Sun Microsystems Plattformen Java SE und EE Verwendung. Teile jeder Plattform sind das Java Development Kit (JDK) sowie die Java Runtime Environment (JRE). Das JDK enthält Werkzeuge, um Java-Programme zu erstellen. Dazu gehört z. B. ein Compiler, welcher den Quellcode in ausführbaren Maschinencode übersetzt. Um Java-Programme ausführen zu können, wird eine JRE benötigt. Sie enthält die Java Virtual Machine (JVM), in welcher Java Programme ablaufen. Für alle gängigen Betriebssysteme existieren Java Virtual Machines, womit die Plattformunabhängigkeit einer Java Software gewährleistet ist (vgl. Abbildung 4.1).

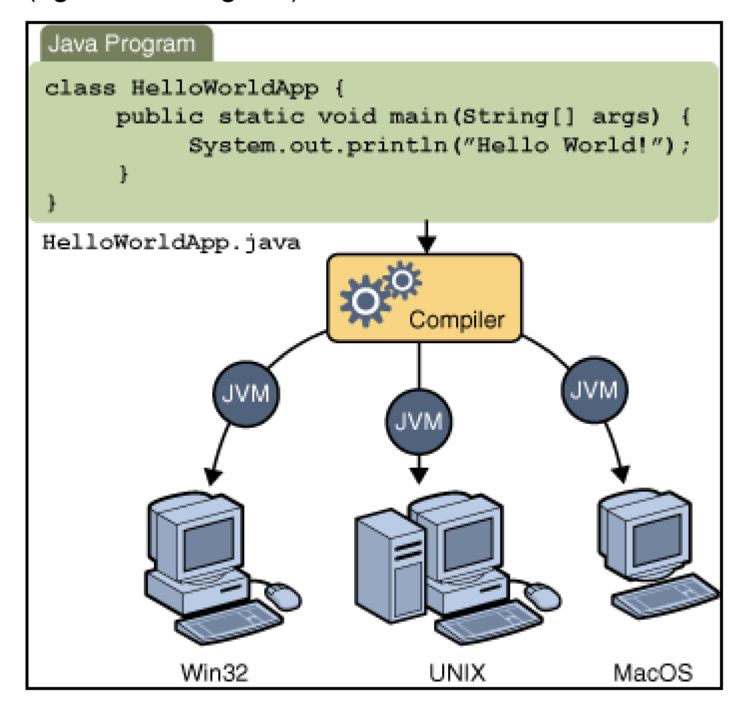


Abbildung 4.1: Plattformunabhängigkeit[9]

[9] Bildquelle: http://java.sun.com/docs/books/tutorial/figures/getStarted/helloWorld.gif

Java ist mithilfe vieler Bibliotheken in seinem Funktionsumfang erweiterbar. Beispielsweise existiert die Schnittstellenspezifikation Java Database Connectivity (JDBC) für Datenbanken. Aufgrund dieser standardisierten und somit austauschbaren Bibliotheken ist ein Datenbankwechsel im Optimalfall ohne Änderungen am Quellcode und lediglich durch den Austausch der JDBC Bibliothek durchführbar.

Java Server Pages (JSP)

Für die Entwicklung des Webfrontends kam die JSP Technologie zum Einsatz. Bei JSP handelt es sich um eine Programmiersprache, die zur Implementierung dynamischer Webseiten geschaffen wurde.

1997 stellte Sun die Servlets-API vor. Servlets sind Java Programme, die auf Webservern ausgeführt werden und Benutzereingaben verarbeiten können. Durch eine Weiterentwicklung entstand im Jahr 1999 JSP. Der Unterschied zwischen Servlets und JSP liegt darin, dass Servlets reine Java-Klassen sind, in die HTML-Code eingebettet wird. Dadurch ist bei Änderungen am Code ein erneutes Übersetzen durch den Compiler notwendig. Bei JSP hingegen wird der Java Code in HTML Dateien eingebettet und muss nicht neu kompiliert werden, damit Änderungen wirksam werden. Dieser Vorteil birgt jedoch auch einen Nachteil. Da der Webserver die JSP Dateien bei jedem Aufruf interpretiert, ist der Seitenaufbau gegenüber der Servlet Technologie langsamer (vgl. SEEBOERGER-WEICHSELBAUM, 2004).

Apache Tomcat

Um JSP Dateien ausführen zu können, wird ein geeigneter Webserver benötigt. Die Farming Cell verwendet den Apache Tomcat[10] in der Version 5.5.25. Dabei handelt es sich um eine freie Software der Apache Software Foundation. Apache Tomcat ist vollständig in Java geschrieben und damit plattformunabhängig einsetzbar. Er beinhaltet den Tomcat-Manager, der eine Übersicht über die installierten Webapplikationen sowie Informationen über den aktuellen Serverstatus liefert. Über den Manager ist es ferner möglich, Webapplikationen komfortabel zu installieren und zu verwalten.

[10] Internetpräsenz: http://tomcat.apache.org

XAMPP
Die frei verfügbare Softwarezusammenstellung XAMPP beinhaltet unter anderem die Softwareprodukte Apache Webserver sowie ein Datenbanksystem[11] bestehend aus der Datenbank MySQL und dem Datenbankmanagementsystem phpMyAdmin. Ebenso sind die Skriptsprachen PHP und Pearl enthalten. Die genannten Produkte sind der Namensgeber: **A**pache, **M**ySQL, **P**HP, **P**earl. Das führende X ist ein Platzhalter für die Betriebssysteme, auf denen die Zusammenstellung installiert werden kann. Die einzelnen XAMPP Distributionen heißen LAMPP (Linux), WAMPP (Windows), MAMPP (Mac OS) (vgl. SEIDLER, 2009). Der entscheidende Vorteil von XAMPP ist die Einfachheit der Installation, denn der Benutzer erhält ein sofort lauffähiges und vorkonfiguriertes System. Die Farming Cell nutzt die Distributionen LAMPP und WAMPP in der Version 1.6.8.

Eclipse
Für die Softwareentwicklung wurde die Integrated Development Environment (IDE) Eclipse[12] in der Version 3.4 verwendet. Bei Eclipse handelt es sich um Open Source Software. Eclipse wurde selbst in Java geschrieben, weswegen sich die IDE durch eine native Java Unterstützung auszeichnet. Der Editor für die Erstellung des Quellcodes gefällt durch Syntax-Highlighting und die halbautomatische Codevervollständigung.
Um auch in anderen Programmiersprachen programmieren und zusätzliche Funktionen verwenden zu können, ist Eclipse durch das Einbinden von Plugins sehr gut erweiterbar. Ein solches Plugin ist Subclipse[13]. Es ermöglicht das Anbinden von Versionsverwaltungsservern wie dem nachfolgend vorgestellten VisualSVN.

VisualSVN
Während eines Softwareentwicklungsprozesses ist die Verwaltung des generierten Quellcodes eine zentrale Herausforderung. Dies ist insbesondere dann der Fall, wenn mehrere Entwickler an einem Projekt arbeiten. Ein Hilfsmittel, um diese Herausforderung zu meistern ist eine Versionsverwaltungssoftware. Im Rahmen des

[11] Ein Datenbanksystem besteht aus den Komponenten Datenbankmanagementsystem (DBMS) und Datenbank (DB). Das DBMS ist eine Verwaltungssoftware und ermöglicht die Administration der Datenbank, welche die eigentlichen Daten hält.
[12] Internetpräsenz von Eclipse: http://www.eclipse.org
[13] Internetpräsenz von Subclipse: http://subclipse.tigris.org

Projektes wurde die freie Software VisualSVN[14] genutzt. Sie ermöglicht es, verteilt generierten Quellcode zusammenzuführen.
Möchte ein Entwickler neuen oder geänderten Quellcode übermitteln, so wird der Code, der zunächst nur auf dem lokalen Rechner des Entwicklers vorhanden ist, in den zentral hinterlegten eingefügt. Andere Entwickler haben nun die Möglichkeit, den neuen Code vom SVN Server in ihre lokale Entwicklungsumgebung zu übernehmen. An den SVN Server übermittelte Änderungen können vom Entwickler mit Kommentaren versehen werden, welche in ihrer Gesamtheit eine Dokumentation des Entwicklungsprozesses darstellen. Jede Änderung besitzt eine eigene Revisionsnummer, ist jederzeit einsehbar und der Entwicklungsprozess somit reproduzierbar.

[14] Internetpräsenz: http://www.visualsvn.com/server

Um den im Rahmen des Projektes Farming Cell entwickelten Quellcode zu verwalten und dauerhaft verfügbar zu halten, wird auf einem virtuellen Server des Rechenzentrums der Universität Hohenheim (vgl. Kapitel 5.4.3) die Software VisualSVN Server betrieben.
Folgende Projekte befinden sich auf dem Server:

Tabelle 4.2: Projekte des VisualSVN Servers

Projektname	Erläuterndes Kapitel
ADIS TINI DLG	5.6.4
ADIS TINI Uni	“
ADIS_TruTest_Client_PC	5.6.2
ADIS_TruTest_TINI	“
Ethernetbox Service	5.6.3
FarmingCell Info Mailer	5.6.7
FarmingCell_Webapplikation	5.6.8
ISOagriNET Parser	5.6.5
Multigasmonitor Service	5.3.4
REST_Service	5.6.10
Schauer Service	5.6.1
agroxml_charge_example	5.2.2
farmingcell_birt	5.6.9.1

Tisan
Die Software mit dem Namen Tisan (vgl. Abbildung 4.2) wurde von der Deutschen Landwirtschafts-Gesellschaft (DLG) entwickelt. Sie wird denjenigen zur Verfügung gestellt, die Geräte oder Software für die durch die DLG durchgeführte ISOagriNET-conform Zertifizierung anmelden.
Tisan ist in der Lage, den Netzwerkverkehr mitzuhören und aufgefangene Nachrichten und Kommunikationsabläufe in Hinblick auf die ISOagriNET Konformität zu überprüfen.

5 Konzeptionierung und Implementierung

Dieses Kapitel stellt Maßnahmen dar, die zum Zwecke der Integration aller in Kapitel 3.2 genannten Anlagen, Sensoren und Verbrauchsmesser in das Gesamtsystem durchgeführt wurden. Darüber hinaus erfolgt die Vorstellung zusätzlich notwendiger, bisher nicht genannter Komponenten.

Beginnend mit Ausführungen zur Gesamtarchitektur der Farming Cell sowie zu den verwendeten Standards ISOagriNET und agroXML, schließen sich komponentenindividuelle Erläuterungen zu den hardwareseitig durchgeführten Maßnahmen an. Den Abschluss des Kapitels bildet die Vorstellung der entwickelten Softwarekomponenten, welche auf den auf Hardwareebene durchgeführten Maßnahmen aufbauen. Um die später separat beleuchteten Komponenten in die Gesamtarchitektur einordnen zu können, wird diese zu Beginn vorgestellt.

5.1 Gesamtarchitektur

Dieses Kapitel gibt einen Überblick über die Hard- und Softwarekomponenten der Farming Cell und deren Beziehungen zueinander, bevor sich Detailbetrachtungen einzelner Komponenten anschließen. Anhand eines Hard- und Softwareschichten-Modells werden alle Komponenten eingeordnet und ihr Zusammenspiel verdeutlicht.

Das Modell (vgl. Abbildung 5.1) besteht aus fünf aufeinander aufbauenden Schichten (engl. Layer). Auf die Untere Hardware Layer, in welcher die Anlagen, Sensoren und Verbrauchsmesser angesiedelt sind, baut die Obere Hardware Layer auf. Dort befinden sich Steuerungskomponenten und solche, die der darunterliegenden Schicht die Kommunikation im Netzwerk erlauben.

Darüberliegend sind Softwarekomponenten in den Schichten Mittlere und Untere Software Layer angeordnet und stellen zum einen Kommunikationspartner der Hardware dar. Zum anderen ist ebendort die Datenhaltung in Form der Datenbank angesiedelt.

Die Oberste Software Layer beheimatet Software, welche dem Menschen oder Fremdsystemen die Interaktion mit der Farming Cell ermöglicht.

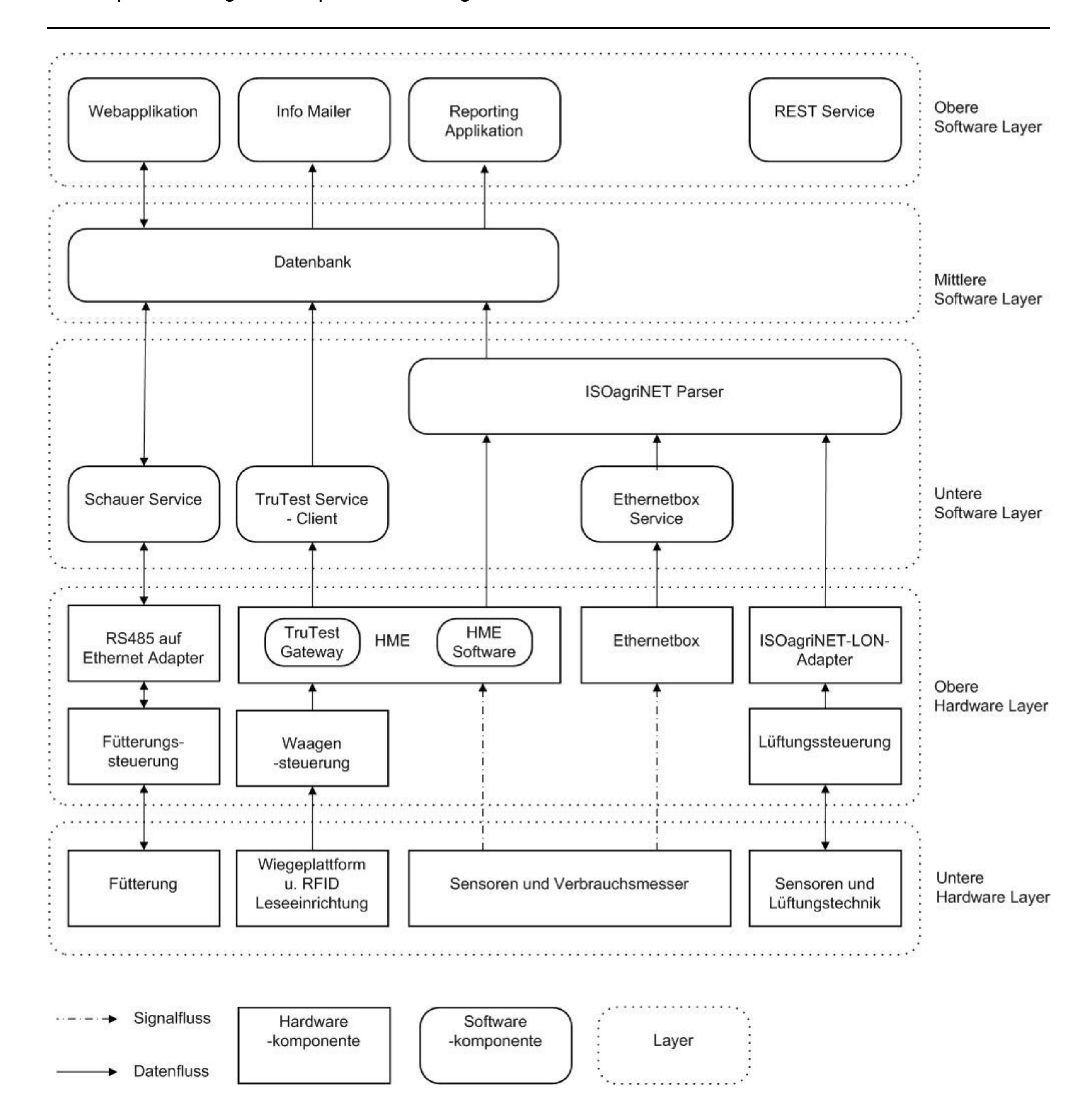


Abbildung 5.1: Hard- und Softwarearchitektur der Farming Cell

Im Folgenden werden die vertikalen, schichtenübergreifend verlaufenden Datenflussstränge vorgestellt und im Zuge dessen das Zusammenspiel involvierter Hard- und Software erläutert. Die Ausführungen dienen dazu, ein Gesamtbild zu zeichnen. Detaillierte Erläuterungen zu einzelnen Hard- und Softwarekomponenten folgen in den sich anschließenden Kapiteln.

Die Stränge, deren oberen Endpunkt jeweils die Datenbank bildet, werden nach ihrer Hardwarebasis unterschieden:

1. Fütterung
2. Wiegeplattform und RFID Leseeinrichtung

3. Sensoren und Verbrauchsmesser
    a. angeschlossen an eine HME
    b. angeschlossen an eine Ethernetbox
4. Sensoren und Lüftungstechnik

Die Steuerungseinheit der **Fütterung**, welche die für den Produktionsprozess relevanten Daten vorhält und Steuerungsbefehle entgegennimmt, ist mithilfe eines Seriell-auf-Ethernet Adapters in das Netzwerk des Versuchsstalls eingebunden. Die Abfrage von in der Datenbank zu speichernden Prozessdaten der Fütterung sowie das Setzen einzelner Parameter übernimmt die Software Schauer Service. Eine von der Firma Schauer bereitgestellte Dynamische Verbindungsbibliothek (engl. Dynamic Link Library, DLL) ist deren Kernstück. Sie ermöglicht die Kommunikation mit der Fütterungssteuerung. Von der Fütterungssteuerung erhaltene Informationen werden in der zentralen Datenbank abgelegt. Ebenso nutzt der Schauer Service Datenbankinformationen um Steuerungsparamter der Fütterung zu manipulieren.

Der **Wiegeplattform** und die **RFID Leseeinrichtung** sind an einer Waagensteuerung angeschlossen. Diese besitzt eine serielle Schnittstelle und ist zum Anschluss an eine Hohenheimer Messwerterfassung, kurz HME, geeignet. Die HME besitzt neben der seriellen auch eine Ethernetschnittstelle (vgl. Kapitel 5.4.1) und beheimatet die Softwarekomponente TruTest Gateway. Sie ist in der Lage, Datensätze der Waagensteuerung abzurufen und an Netzwerkteilnehmer weiterzugeben. Das TruTest Gateway nimmt Anfragen des TruTest Service - Clients entgegen, reicht sie an die Waagensteuerung weiter und liefert den erhaltenen Antwortdatensatz zurück. Der TruTest Service - Client schreibt im Anschluss die erhaltenen Wiegedaten in die Datenbank.

Sensoren und Verbrauchsmesser sind an die **Hohenheimer Messwerterfassung** (vgl. Kapitel 5.4.1) anschließbar. Die Abfrage ihrer Messwerte erfolgt durch die HME Software, welche darüber hinaus deren Publikation im Netzwerk übernimmt. Die Software ISOagriNET Parser fängt die publizierten Nachrichten auf und legt die enthaltene Information in der Datenbank ab.
Messwerte von **Sensoren und Verbrauchsmessern**, die an eine **Ethernetbox** (Ethernetbox der Firma better networks, vgl. Kapitel 5.4.2) angeschlossen sind,

werden netzwerkbasiert von der Software Ethernetbox Service abgefragt und ebenfalls publiziert. Der ISOagriNET Parser legt die empfangenen Informationen in der Datenbank ab.

Die Lüftungssteuerung mit angeschlossenen **Sensoren und Lüftungstechnik** ist durch einen nicht selbstentwickelten ISOagriNET-LON-Adapter mit dem Netzwerk verbunden. Auch dieser Adapter publiziert Werte im Netzwerk, welche vom ISOagriNET Parser aufgenommen und in die Datenbank geschrieben werden.

Die in Abbildung 5.1 genannten Softwarekomponenten befinden sich auf verschiedenen Computern. Diese und mögliche Konfigurationen werden nachfolgend ebenso vorgestellt (vgl. Tabelle 5.10), wie die erwähnten Anlagen, Sensoren, Verbrauchsmesser und weitere Komponenten.

5.2 Standards

Die Verwendung der zwei Standards ISOagriNET und agroXML ist das zentrale Leitbild der Farming Cell. Während ISOagriNET für die betriebsinterne Kommunikation zum Einsatz kommt, ist die betriebsexterne Datenbereitstellung mit agroXML realisiert. Neben ihrem Anwendungsbereich unterscheiden sich die beiden Standards auch hinsichtlich ihres Definitionsumfanges. Im Gegensatz zu ISOagriNET, welches Datenformate und Kommunikationsabläufe definiert, beschreibt agroXML, als auf XML basierende Datenaustauschsprache, lediglich die Struktur der Daten, nicht jedoch die Art und Weise des Austausches (vgl. MARTINI 2007 und Kapitel 5.2.2).

5.2.1 ISOagriNET

Die Analyse der Gegebenheiten und der daraus ermittelten Anforderungen hat ergeben, dass ein Teil der auszutauschenden Daten mittels bereits vorhandener ADED Entitäten modelliert werden kann. Einige Items, insbesondere die Werte der Verbrauchsmesser, sind hingegen nicht im gegenwärtig aktuellen Data Dictionary Agro2010[15] enthalten. Aus diesem Grunde wurden zwei Entitäten sowie zehn Items neu definiert. Dieses Vorgehen ist standardkonform (vgl. ISO, 2009), da die Entitäten lediglich innerhalb der Farming Cell Anwendung finden.

[15] Data Dictionaries bereitgestellt durch das LKV NRW: http://www.lkv-wl.de/index.php?id=309. Abrufdatum 05.02.2010

Auf den folgenden Seiten sind alle im System der Farming Cell verwendeten Entitäten samt deren Items genannt. Der Aufbau und die Inhalte der Tabellen sind in Auszügen (Stand 30.09.2009) von den Seiten des LKV NRW[16] übernommen. Lücken- oder fehlerhafte Datensätze blieben unverändert.

Erläuterungen zu den Spaltenbezeichnern und Inhalten:

ID	-	fortlaufende Nummer.
TYP	-	Indikator ob es sich um den Primärschlüssel (PK) der Tabelle handelt (KEY), ob das Feld optional ist, d.h. nicht gesetzt sein muss (OPT) oder ob es zwingend (mandatory) zu füllen ist (MAN).
Nr	-	Nummer des Items.
Name	-	Name des Items.
Beschreibung	-	Erläuterung zu Inhalt/Zweck des Items.
Datentyp	-	AN oder N – alphanumerisch oder numerisch.
Länge	-	Gesamtlänge des Feldes (inkl. Nachkommastellen).
Auflösung	-	Anzahl der Nachkommastellen.
Einheit	-	Bei Datums- und Uhrzeit Feldern die jeweilige Formatierung.
Codeset	-	Referenz ID des Codesets; 0 wenn kein Codeset benutzt wird.

[16] LKV NRW: Landeskontrollverband Nordrhein-Westfalen, Internetpräsenz: http://www.lkv-wl.de

Folgende Entitäten des Data Dictionaries Agro2010 fanden Verwendung:

Bereich Übergreifendes (Nummernkreis 100000 - 199999)

Entity 101000 – KlimaLuftdaten

Tabelle 5.1: ADED Entität 101000

ID	Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Code-set
1	KEY	901002	location	Location laut ISO17532: BetrNr:Stall.Abteil.Bucht	AN	40	0	location type	0
2	MAN	901013	timestamp		N	17	0	CCYYMM DDHH mmSSsss	0
3	MAN	101000	Temperatur	Aktuelle Temperatur in ??C	N	6	2		0
4	OPT	101001	Feuchte	Aktuelle relative Luftfeuchte in %	N	4	1	%	0
5	OPT	101005	Kohlendioxid	Aktueller CO2 Gehalt der Luft	N	6	0	ppm	0
6	OPT	101006	Ammoniak	Aktueller NH3 Gehalt der Luft	N	6	1	ppm	0

Entity 101001 - KlimaAussen

Tabelle 5.2: ADED Entität 101001

ID	Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Code-set
1	KEY	901002	location	Location laut ISO17532: BetrNr:Stall.Abteil.Bucht	AN	40	0	location type	0
2	MAN	901013	timestamp		N	17	0	CCYYMM DDHHmm SSsss	0
3	MAN	101000	Temperatur	Aktuelle Temperatur in ??C	N	6	2		0
4	OPT	101001	Feuchte	Aktuelle relative Luftfeuchte in %	N	4	1	%	0
5	OPT	101002	Wind	Aktuelle Windgeschwindigkeit in m/s	N	4	2	m/s	0

ID	Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Code-set
6	OPT	101003	Windrichtung	Aktuelle Windrichtung	N	4	1		0
7	OPT	101014	Luftdruck	Aktueller Luftdruck in hPa	N	4	0	hPa	0
8	OPT	101004	Lichtstärke	Beleuchtungsst??rke in Lux	N	6	0	Lux	0
9	OPT	101015	Regen	Regenmelder	N	1	0	Codeset	8888
10	OPT	101016	Enthalpie	Berechneter Enthalpiegehalt der Luft in kJ / kg tr.Luft	N	6	2	kJ/kg	0

Entity 101005 – KlimaTierdaten

Tabelle 5.3: ADED Entität 101005

ID	Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Code-set
1	KEY	901002	location	Location laut ISO17532: BetrNr:Stall.Abteil.Bucht	AN	40	0	location type	0
2	MAN	901013	timestamp		N	17	0	CCYYMM DDHHmm SSsss	0
3	OPT	101008	Tierart	Tierart	N	2	0	Codeset	1101
4	OPT	620000	Alter	Durchschnittsalter in Tagen	N	4	0	Tage	0
5	OPT	620001	Gewicht	Durchschnittsgewicht pro Tier in Kilogramm	N	6	2	Kg	0
6	OPT	663502	S_ANZ_G2		N	8	0		0
7	OPT	101029	Wasser-verbrauch	Wasserverbrauch am aktuellen Tag	N	6	0	Liter	0

Bereich Schweine (Nummernkreis 600000 - 699999)

Entity 610011 – WIEGEN

Tabelle 5.4: ADED Entität 610011

ID	Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Codeset
1	KEY	610176	TIER_ID	Individuelle elektronische Tiernummer	N	15	0		0
2	KEY	610076	WIEGEDATUM		N	8	0	ccyymmdd	0
3	OPT	610077	WIEGEZEIT		N	6	0	hhmmss	0
4	OPT	610035	GEWICHT		N	4	1		0
5	OPT	610078	GEWICHTSART	Tot-, Lebendgewicht	N	1	0		6078
6	OPT	610079	GEWICHTSTYP	einzeln oder in Gruppe gewogen oder geschätzt	N	1	0		6079

Folgende zehn Items wurden neu definiert

Tabelle 5.5: Farming Cell interne ADED Items

Typ	Nr	Name	Beschreibung	Datentyp	Länge	Auflösung	Einheit	Codeset
OPT	1	coordinate_x	x Koordinate des Messpunktes	N	4	2	m	0
OPT	2	coordinate_y	y Koordinate des Messpunktes	N	4	2	m	0
OPT	3	coordinate_z	z Koordinate des Messpunktes	N	4	2	m	0
OPT	4	ch4	Aktueller CH4 Gehalt der Luft	N	6	1	ppm	0
OPT	5	n2o	Aktueller N2O Gehalt der Luft	N	6	1	ppm	0
OPT	6	electric_meter	Verbrauch seit letzter Messung	N	8	2	kWh	0
OPT	7	water_meter	Verbrauch seit letzter Messung	N	8	2	ml	0
OPT	8	heat_meter	Verbrauch seit letzter Messung	N	8	2	kWh	0

OPT 9	gas_meter	Verbrauch seit letzter Messung	N	8	2	m3	0
OPT 10	air_speed	Luftgeschwindigkeit	N	3	1	m/s	0

Folgende zwei Entitäten wurden, aufbauend auf den vorhandenen sowie den neu erstellten Items, definiert.

Entity 1 - environment_values

Tabelle 5.6: ADED Entität 1

ID	Typ	Nr	Name	Beschreibung	Daten-typ	Länge	Auflös-ung	Einheit	Code-set
1	MAN	901002	location	Location laut ISO17532: BetrNr:Stall.Abteil.Bucht	AN	40	0	location type	0
2	MAN	901013	timestamp		N	17	0	CCYYMM DDHHmm SSsss	0
3	OPT	1	coordinate_x	x Koordinate des Messpunktes	N	4	2	m	0
4	OPT	2	coordinate_y	y Koordinate des Messpunktes	N	4	2	m	0
5	OPT	3	coordinate_z	z Koordinate des Messpunktes	N	4	2	m	0
6	OPT	101000	Temperatur	Aktuelle Temperatur in Grad Celsius	N	6	2		0
7	OPT	101001	Feuchte	Aktuelle relative Luftfeuchte in %	N	4	1	%	0
8	OPT	101005	Kohlendioxid	Aktueller CO2 Gehalt der Luft	N	6	0	ppm	0
9	OPT	101006	Ammoniak	Aktueller NH3 Gehalt der Luft	N	6	1	ppm	0
10	OPT	4	ch4	Aktueller CH4 Gehalt der Luft	N	6	1	ppm	0
11	OPT	5	n2o	Aktueller N2O Gehalt der Luft	N	6	1	ppm	0
12	OPT	101019	Unterdruck	Aktueller Unterdruck im Abteil	N	4	0	Pa	0
13	OPT	101004	Helligkeit	Beleuchtungsstärke in Lux	N	6	0	Lux	0

ID	Typ	Nr	Name	Beschreibung	Daten-typ	Länge	Auflös-ung	Einheit	Code-set
14	OPT	10	air_speed	Luftgeschwindigkeit	N	3	1	m/s	0

Entity 2 - meter_values

Tabelle 5.7: ADED Entität 2

ID	Typ	Nr	Name	Beschreibung	Daten-typ	Länge	Auflös-ung	Einheit	Code-set
1	MAN	901002	location	Location laut ISO17532: BetrNr:Stall.Abteil.Bucht	AN	40	0	location type	0
2	MAN	901013	timestamp		N	17	0	CCYYMM DDHHmm SSsss	0
3	OPT	1	coordinate_x	x Koordinate des Messpunktes	N	4	2	m	0
4	OPT	2	coordinate_y	y Koordinate des Messpunktes	N	4	2	m	0
5	OPT	3	coordinate_z	z Koordinate des Messpunktes	N	4	2	m	0
6	OPT	6	electric_meter	Verbrauch seit letzter Messung	N	8	2	kWh	0
7	OPT	7	water_meter	Verbrauch seit letzter Messung	N	8	2	ml	0
8	OPT	8	heat_meter	Verbrauch seit letzter Messung	N	8	2	kWh	0
9	OPT	9	gas_meter	Verbrauch seit letzter Messung	N	8	2	m3	0

Die neuen Entitäten wurden in die SQLite Datenbank der Software Tisan (vgl. Kapitel 4.2) eingepflegt. Auf diese Weise stehen sie für mit Tisan durchgeführte Tests zur Verfügung. Die dateibasierte Datenbank findet auch Anwendung in einigen der in Kapitel 5.6 vorgestellten Softwareentwicklungen.

5.2.2 agroXML

Kapitel 2 hat eine Einführung in agroXML gegeben und dargestellt, welchem Zweck XML-Schema dient. An diesem Punkt wird an dieser Stelle angeknüpft. Es folgen Ausführungen, welchen Sachverhalt die durch das KTBL im Rahmen von IT FoodTrace entwickelten Schema Dateien modellieren. Daran anschließend wird erläutert, wie im konkreten Fall der Farming Cell mithilfe selbstentwickelter Software und unter Verwendung des Modells des KTBL, XML Dateien mit realen Betriebsdaten erzeugt wurden.

XML Schema des KTBL

Da die Farming Cell ihre Daten zentral vorhält, sind einzeltierbezogene Informationen wie Tiergewichte verfügbar. Diese können für Dritte, beispielsweise einen Schlachthof, von Interesse sein. Seitens des KTBL wurde ein Schema definiert, das folgenden Sachverhalt abbildet.

Mehrere Einzeltiere werden zu einer Gruppe (Charge) zusammengefasst. Diese in Form einer XML Datei vorliegende Charge besitzt ein Gesamtgewicht und eine eindeutige ID. Ferner enthält die Datei Verweise (URI) auf Informationen zu jedem einzelnen Tier und auf deren gemeinsamen Ursprungsbetrieb (Farm). Der Ursprungbetrieb wird in einer separaten XML Datei durch Name, Adresse und weitere Attribute eindeutig beschrieben. Die ebenfalls in Form einzelner XML Dateien modellierten Tiere besitzen die Eigenschaften Ohrmarkennummer, Geschlecht, Rasse sowie ein Gewicht mit dazugehörigem Wiegedatum.
Tabelle 5.8 nennt die durch das KTBL bereitgestellten XML Schema Dateien (XSD) und deren Inhalt. Jede dieser Dateien enthält Vorgaben über die zulässigen Inhalte und den Aufbau der zugehörigen XML Dateien.

Tabelle 5.8: XML Schema Dateien des KTBL

Schema Datei	Inhalt
Address.xsd	Kontaktdaten
agroxml.xsd	Namespaces und Modulimporte (agroxml.xsd ist Root Element einer Instanz)
Animal.xsd	Für alle Tierarten gültige Eigenschaften
AnimalEggs.xsd	Speziell auf Legehennen zutreffende Eigenschaften

AnimalMeat.xsd	Speziell auf Schweine zutreffende Eigenschaften
AnimalMilk.xsd	Speziell auf Milch gebende Tiere zutreffende Eigenschaften
Charge.xsd	Eigenschaften einer Tiercharge
CommonBasic Components.xsd	Erläuterungen zu Datentypen
CoreComponent Types.xsd	Einheiten etc. von Datentypen
Event.xsd	Typ und Eigenschaften eines Ereignisses
Farm.xsd	Eigenschaften eines Betriebes
Gml.xsd	Datencontainerkonstrukten wie Listen etc.
Stall.xsd	Eigenschaften eines Stalls
Xlinks.xsd	Eigenschaften eines XLinks

Die für den abzubildenden Sachverhalt wichtigsten Dateien sind Charge.xsd, Farm.xsd und AnimalMeat.xsd, denn sie definieren die Strukturen der zu modellierenden Objekte Tiercharge, landwirtschaftlicher Betrieb und Einzeltier.
Um aus den bereitgestellten XSD Dateien XML Dateien erzeugen zu können, wurde das Framework Maven[17] verwendet. Maven setzt auf Java Architecture for XML Binding (JAXB) auf und ist in der Lage, aus XSD Dateien Java Klassen zu erzeugen, welche anschließend als Basis für die Erzeugung von XML Dateien verwendet werden können. Ein entsprechendes Projekt mit dem Namen agroXML_charge_example befindet sich im Repository des VisualSVN Servers (vgl. Kapitel 4.2). Es beinhaltet neben selbstgeschriebenen Java Klassen (siehe unten) die durch das KTBL bereitgestellten und oben genannten 14 Schema Dateien sowie die daraus generierten 139 Java Klassen.

Beispiel: Inhalt der XML Schema Datei AnimalMeat.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Copyright Â© 2007 Kuratorium für Technik und Bauwesen in der Landwirtschaft
e.V. (KTBL). All Rights Reserved. http://www.agroxml.de/Legal/
...
-->
<xsd:schema xmlns="http://www.agroxml.de/schema/devel"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.agroxml.de/schema/devel"
elementFormDefault="qualified" attributeFormDefault="unqualified">
      <xsd:include schemaLocation="CommonBasicComponents.xsd"/>
```

[17] Internetpräsenz: http://maven.apache.org

```
      <xsd:include schemaLocation="Animal.xsd"/>
      <xsd:element name="Pig" type="PigType">
            <xsd:annotation>
                  <xsd:documentation xml:lang="de">Dieses Element wird in
der Instanz nie verwendet. Anstattdessen werden die konkreten in der
substitutionGroup abgelegten Elemente (Milchkuh, Schwein...)
eingebunden.</xsd:documentation>
            </xsd:annotation>
      </xsd:element>
      <xsd:complexType name="PigType">
            <xsd:annotation>
                  <xsd:documentation>In diesem Datentyp sind alle fÃ¼r die
vorhandenen Tierarten gemeinsamen Elemente
untergebracht.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexContent>
                  <xsd:extension base="AbstractAnimalType">
                        <xsd:sequence>
                              <xsd:element name="EartagNumber"
type="xsd:token" minOccurs="0"/>
                              <xsd:element name="PigRace" type="CodeType"
minOccurs="0"/>
                        </xsd:sequence>
                  </xsd:extension>
            </xsd:complexContent>
      </xsd:complexType>
</xsd:schema>
```

Beispiel: Inhalt der mit Maven erzeugten Java Klasse PigType.java

```
package agroxml.generated;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlType;
import javax.xml.bind.annotation.adapters.CollapsedStringAdapter;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "PigType", propOrder = {
    "eartagNumber",
    "pigRace"
})

public class PigType
    extends AbstractAnimalType
{
    @XmlElement(name = "EartagNumber")
    @XmlJavaTypeAdapter(CollapsedStringAdapter.class)
    protected String eartagNumber;
    @XmlElement(name = "PigRace")
    protected CodeType pigRace;

    /**
     * Gets the value of the eartagNumber property.
     *
     * @return
     *     possible object is
```

```
     *     {@link String }
     */
    public String getEartagNumber() {
        return eartagNumber;
    }

    /**
     * Sets the value of the eartagNumber property.
     *
     * @param value
     *     allowed object is
     *     {@link String }
     */
    public void setEartagNumber(String value) {
        this.eartagNumber = value;
    }

    /**
     * Gets the value of the pigRace property.
     *
     * @return
     *     possible object is
     *     {@link CodeType }
     */
    public CodeType getPigRace() {
        return pigRace;
    }

    /**
     * Sets the value of the pigRace property.
     *
     * @param value
     *     allowed object is
     *     {@link CodeType }
     */
    public void setPigRace(CodeType value) {
        this.pigRace = value;
    }
}
```

Das Erzeugen von XML Dateien konkreter Tierchargen, Betriebe und Einzeltiere ist mittels der eigens implementierten Klassen

- ChargeCreator.java,
- FarmCreator.java und
- PigCreator.java

möglich. Die Inhalte der zu erzeugenden XML Datei werden an die entsprechende Creator Klasse übergeben. Ein Beispiel (Klasse Test.java) mit Beispieldaten zum Erzeugen von XML Dateien eines Betriebes, einer Charge und mehrerer Einzeltiere ist im Projekt agroXML_charge_example des VisualSVN Servers enthalten.

Beispiel: XML Datei eines Einzeltiers

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<pigType  id="969000000032852"  xmlns="http://www.agroxml.de/schema/devel"
xmlns:xlink="http://www.w3.org/1999/xlink">
 <Events>
  <Weighing>
   <DateOrTimestamp>2008-09-12</DateOrTimestamp>
   <Weight uom="kg">111.0</Weight>
  </Weighing>
 </Events>
 <Sex>m</Sex>
 <EartagNumber>969000000032852</EartagNumber>
 <PigRace>PIxDL</PigRace>
</pigType>
```

Über die Erzeugung der XML Dateien hinaus bietet das Framework ebenso die Möglichkeit, derartige Dateien wiedereinzulesen und für eine Weiterverarbeitung in Java Objekte abzulegen. Somit ist das Framework als Zwischenschicht für Projekte wie beispielsweise einen Webservice zur Bereitstellung und Entgegennahme von agroXML Daten einsetzbar.
Die Verwendung des Frameworks mit Daten der Farming Cell Datenbank ist möglich, die entsprechende Datenbankschnittstelle jedoch nicht implementiert.

5.3 Anlagen

In diesem Kapitel werden die Möglichkeiten der ISOagriNET konformen Anbindung der im Versuchsstall befindlichen Komponenten an das Stallnetz geprüft und geeignete Wege vorgestellt. Dies umfasst die Fütterungs- und die Lüftungsanlage sowie die Tierwaage mit RFID Leseeinrichtung und den Multigasmonitor.

5.3.1 Fütterungsanlage

Im Rahmen der Vorstellung der Fütterungsanlage in Kapitel 3.2.1.1 wurde die Ausstattung ihrer Steuerungseinheit (nachfolgend Fütterungscomputer genannt) mit Hardwareschnittstellen dargestellt. Es ist lediglich eine serielle RS422 Schnittstelle für Kommunikationszwecke vorhanden. Eine Umsetzung dieser auf das durch ISOagriNET vorgegebene RJ45 (Ethernet) ist mittels eines seriellen Servers, wie ihn Abbildung 5.2 zeigt, erfolgt.



Abbildung 5.2: Serielle Server JetPort 5601 der Firma Korenix

Alternativ zu der Verwendung eines seriellen Servers, wäre die Anbindung des Fütterungscomputers an das Ethernet durch Anschluss an einen Computer möglich gewesen, welcher seinerseits über eine Netzwerkschnittstelle verfügt. Diese Variante hätte jedoch das Verlegen eines zusätzlichen seriellen Kabels oder die Installation eines Computers in unmittelbarer Nähe des Fütterungscomputers bedeutet. Durch die Verwendung eines seriellen Servers hingegen, konnte die im Stall bereits vorhandene Ethernet Infrastruktur genutzt werden.
Die für die ISOagriNET konforme Kommunikation mit der Fütterungssteuerung entwickelte Software stellt Kapitel 5.6.1 vor.

5.3.2 Lüftungsanlage

Die im Versuchsstall genutzten Klimacomputer der Firma Möller Agrarklima GmbH verwenden den Feldbus Local Operating Network (LON) für die Kommunikation. Die Umsetzung auf Ethernet erfolgt mittels eines ISOagriNET-LON-Adapters, der von der Firma Möller entwickelt wurde (vgl. Abbildung 5.3 und Anonymus, 2009b).



Abbildung 5.3: ISOagriNET-LON-Adapter
Bildquelle: Möller Agrarklima GmbH

Der als ISOagriNET konform zertifizierte Adapter (DLG, 2009) ist in der Lage, ADIS/ADED Nachrichten mit stallklimarelevantem Inhalt zu publizieren (UDP Multicast). Inhalt (ADED Entität) und Sendeintervall der Nachrichten sind konfigurierbar (vgl. Anonymus, 2009b).

5.3.3 Waage mit RFID Reader

Abbildung 5.4 zeigt das Waagenterminal XR3000 der Firma Tru-Test, an welche die Wiegeeinheit sowie die nachfolgend vorgestellte RFID Leseeinheit angeschlossen sind. Auf generierte Wiegedatensätze (RFID Ohrmarkennummer und dazugehöriges Gewicht) kann über eine der beiden RS232 Schnittstellen der XR3000 zugegriffen werden. Das Kommunikationsprotokoll des Waagenterminals ist frei verfügbar (vgl. Anonymus, 2004a).



Abbildung 5.4: Waagenterminal Tru-Test XR3000

Nach dem gleichen Prinzip wie die serielle Schnittstelle des Fütterungscomputers auf Ethernet umgesetzt wurde (serieller Server), hätte auch die **Steuereinheit der Waage** an das Ethernet angebunden werden können. Es wurde jedoch ein anderer Weg beschritten und die **Hohenheimer Messwerterfassung** (vgl. Kapitel 5.4.1) als Gateway verwendet. Diese verfügt über eine geeignete serielle Schnittselle und realisiert die Ethernetanbindung des Waagenterminals. Die für die Kommunikation zwischen dem Waagenterminal und Netzwerkteilnehmern entwickelte Software der HME wird in Kapitel 5.6.2 vorgestellt.

Bei der eingesetzten **RFID Leseeinrichtung** handelt es sich um den Langstreckenleser ASR700 (vgl. Anonymus, 2008) mir einer Antenne des Typs ASA009 (vgl. Anonymus, 2006) der Firma Agrident GmbH. Der Anschluss an einen Computer für Konfigurationszwecke ist über eine RS232 oder RS485 Schnittstelle möglich.

Die Einbindung der RFID Leseeinrichtung in das Stallnetz erfolgt mithilfe der Waagensteuerung, welche über zwei Wiegezellenanschlüsse verfügt. Diese eignen sich zum Anschluss, so dass die Übertragung der ID im Lesefeld befindlicher Transponder und deren Zuordnung zu Einzeltiergewichten in der Waagensteuerung erfolgt. Diese Daten sind mithilfe der HME für alle Netzwerkteilnehmer ISOagriNET konform zugänglich.

5.3.4 Multigasmonitor

Der in Kapitel 3.2.1.4 vorgestellte Multigasmonitor muss über eine serielle Schnittstelle (RS232) mit einem Computer verbunden werden, damit seine Steuerung möglich und die persistente Speicherung der Messwerte gewährleistet ist. Die Option, eine eigene Software zu entwickeln, die eine direkte Kommunikation mit dem Multigasmonitor und die anschließende ISOagriNET konforme Publikation implementiert, bestand aufgrund der proprietären Schnittstelle des Multigasmonitors nicht.

Die durch die Herstellerfirma bereitgestellte **Steuerungssoftware** legt die vom Multigasmonitor übermittelten Messwerte in einer eigenen Datenbank ab. Bei der Datenbank handelt es sich um Microsoft SQL Server[18], in welcher auch die Konfigurationsparameter, unter anderem Messstellen und –zeitpunkte, hinterlegt sind. Eine Dokumentation des Datenbankschemas stand nicht zur Verfügung, sein Studium jedoch hat ausreichende Erkenntnisse geliefert, um eine prototypische Implementierung einer Software mit dem Namen **Multigasmonitor Service** vorzunehmen. Ihre Aufgabe ist es, die in der Microsoft SQL Server Datenbank gesammelten Messwerte regelmäßige in die MySQL Datenbank der Farming Cell zu übertragen. Sie folgt mit einer dateibasierten Konfiguration und Logging Mechanismen dem Prinzip aller in Kapitel 5.6 vorgestellten Softwarekomponenten. Auf eine detaillierte Vorstellung wird im Rahmen dieser Arbeit verzichtet, da sie sich in einem frühen Stadium befindet und nicht erprobt wurde. Die Software ist im SVN Repository verfügbar.

5.4 Computer und Gateways

Das Ziel, eine permanente Erfassung und Verfügbarkeit der Prozessdaten zu erreichen, macht die Verwendung zusätzlicher Hardwarekomponenten notwendig. Diese müssen zum einen den Zugang zu den datenproduzierenden Anlagen, Sensoren und Verbrauchsmessern ermöglichen. Zum anderen sind eine zentrale Datenbank für die Daten, sowie geeignete Plattformen wie Computer oder Mikroprozessorsysteme zum Ausführen notwendiger Dienste erforderlich. Die folgenden Abschnitte stellen diese Hardwarekomponenten vor.

[18] Internetpräsenz: http://www.microsoft.com/germany/sql/2008/default.mspx

5.4.1 Hohenheimer Messwerterfassung (HME)

Die Anforderung, Sensoren in das Gesamtsystem zu integrieren und die Datenübertragung deren Messwerte ISOagriNET konform zu realisieren, führte zu der Entwicklung eines eigenen Systems. Es trägt den Namen Hohenheimer Messwerterfassung, kurz HME. Die HME ermöglicht den Anschluss analoger Sensoren für die Erfassung von Differenzdruck, NH_3 und CO_2 Konzentration, Temperatur, Helligkeit und Luftfeuchtigkeit. Die Entscheidung für eine Neuentwicklung kam zustande, da keine Produkte mit hinreichender Funktionalität am Markt verfügbar waren.



Abbildung 5.5: Hohenheimer Messwerterfassung

Als Basis für die Entwicklung wurde die Mikroprozessorplattform TINI[19] der Firma Maxim verwendet, die sich insbesondere durch ihr gutes Preis-Leistungs-Verhältnis, ihre Programmierbarkeit in Java, geeignete Schnittstellen für die Netzwerkkommunikation und die Möglichkeit auszeichnet, Messmodule anschließen zu können (KUHLMANN et al., 2009). Die obige Abbildung zeigt das entwickelte System. Es umfasst den Mikroprozessor TINI auf einer Schnittstellenkarte (DSTINIs400 von Maxim), einen darunter befindlichen Wireless LAN Adapter und ein Netzteil.

[19] Das tiny Internet interface (TINI) von Maxim ist ein mit Java programmierbares Mikroprozessor-Board. http://www.maxim-ic.com/products/microcontrollers/tini/

Die für den Anschluss der Sensoren genutzte Schnittstelle ist der One-Wire Bus, dem eine Linientopologie zugrunde liegt. Sogenannte One-Wire Module, wie sie Abbildung 5.6 zeigt, sind mittels Patchkabel bis zu einer Länge von 15m in Reihe schaltbar. Die Anzahl anschließbarer Module ist theoretisch unbegrenzt (KUHLMANN et al., 2009). Die Spannungsversorgung angeschlossener Sensoren ist durch die Einspeisung einer Versorgungsspannung mithilfe eines Zwischensteckers an beliebiger Stelle des Busses möglich.





Abbildung 5.6: One-Wire Module
Links: Temperatur/Volt-Modul der Firma iButtonLink; Rechts: 4 Kanal A/D Modul (Eigenbau)

Neben dem abgebildeten Modul mit vier Spannungseingängen (Abbildung 5.6 rechts) wurde auch eines zum Zählen digitaler Pulse entwickelt. Es sollte für den Anschluss von Stromenergiezählern oder Wasservolumenmessern genutzt werden. Da dessen Test nicht zufriedenstellend verlief und die Produktion des zentralen Zählerbausteins ersatzlos eingestellt wurde, ist es nicht im produktiven Einsatz. Eine Alternative Pulserfassungsmethode war in Form der Ethernetbox (vgl. Kapitel 5.4.2) vorhanden.

In der Farming Cell finden One-Wire Module des Herstellers iButtonLink LLC[20] für die Erfassung folgender Messwertarten Verwendung:

- Temperatur: - 40 bis 85 °C
- Helligkeit: 0 bis 100 %; die Intensität einer 100 Watt Glühbirne in einer Entfernung von 15,24 cm entsprechen 100 %, 0 % entspricht totaler Finsternis (Anonymus, 2009c)
- Relative Luftfeuchte: 0 - 100 %
- Spannung: 0 bis 10 Volt
- Strom: 0 bis 20 Ampere

[20] Internetpräsenz: http://www.ibuttonlink.com

Auf Grund der möglichen Signalarten Spannung und Strom ist eine Vielzahl Sensoren an die HME anschließbar, wodurch die Voraussetzung zur umfassenden und standardisierten Datenerfassung im landwirtschaftlichen Betrieb geschaffen wurde (KUHLMANN et al., 2009).

Über die Möglichkeit hinaus, Sensoren an die HME anzuschließen, ist ihre RS232 Schnittstelle für die Kommunikation mir anderen Geräten geeignet. Im vorliegenden Fall wurde das Waagenterminal (vgl. Kapitel 5.3.3) an die HME angeschlossen.

Die für die Kommunikation mit dem Waagenterminal sowie die für die Messwerterfassung und -publikation entwickelten Softwarekomponenten werden in den Kapiteln 5.6.2 und 5.6.4 vorgestellt.

Im Praxiseinsatz hat sich die HME bewährt. Sie ist in zweifacher Ausführung im Versuchsstall der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof im Einsatz. Einmal aufgesetzt arbeitet sie wartungsfrei. Ihre Bedienung geschieht browserbasiert und die Konfiguration während des laufenden Betriebes ist möglich. Die Leistungsaufnahme beträgt maximal 2,5 Watt, die Kosten für die Komponenten der HME exklusive WLAN Modul belaufen sich auf 136,85 € brutto (KUHLMANN, et al., 2009).

Die Hohenheimer Messwerterfassung wurde durch die DLG als ISOagriNET konform zertifiziert (DLG, 2009).

5.4.2 Ethernetbox

Die Ethernetbox der Firma better networks[21] basiert auf einem Mikroprozessor der amerikanischen Firma Rabbit. Sie ermöglicht den Anschluss von Sensoren (Strom- oder Spannungssignal) und Verbrauchsmessern (S0 Signal) und sie bietet netzwerkbasierte Abfragemöglichkeiten für die Signale angeschlossener Komponenten sowie ihre internen Zählerstände. Abbildung 5.7 zeigt eine solche Ethernetbox.

[21] Internetpräsenz: http://www.betternetworks.de

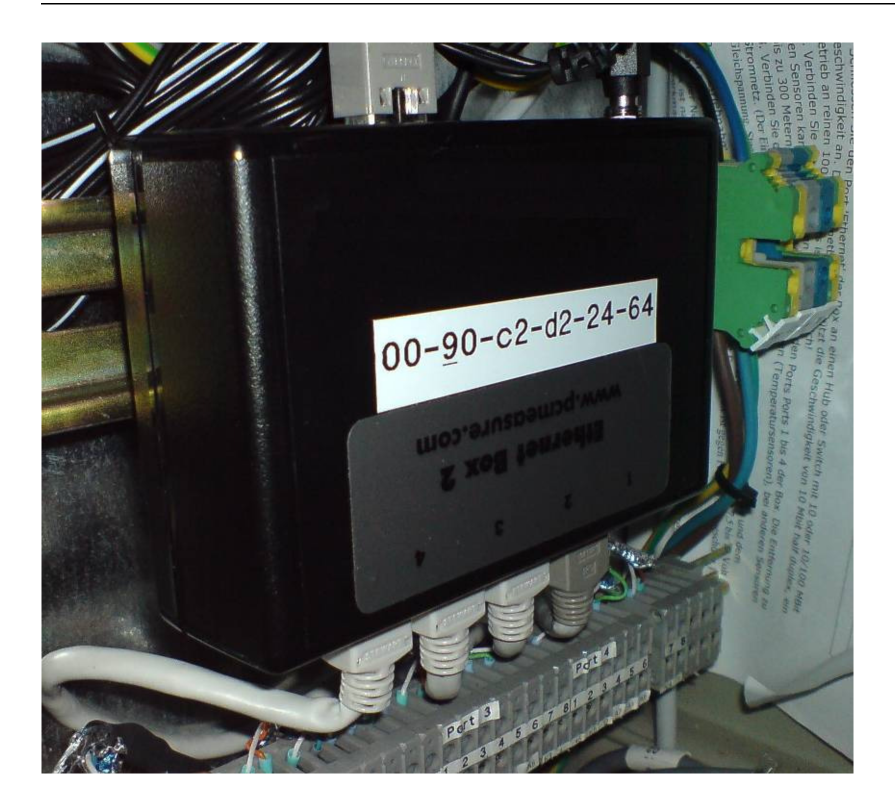


Abbildung 5.7: Ethernetbox

Das System dient seiner ursprünglichen Bestimmung nach der Überwachung von Serverräumen und ist in zwei Versionen verfügbar, die sich hinsichtlich der Schnittstellenausstattung unterscheiden. Version 1 verfügt über 12 digitale Eingänge, Version 2 ist darüber hinaus mit 8 analogen Eingängen ausgestattet. Anliegende Spannungen können im Bereich von 0 bis 10 Volt mit einer Genauigkeit von 0,1 mV erfasst werden. Die Zählereingänge sind in der Lage bis zu 100 Impulse in der Sekunde (spezielle Firmware Version, Standard ist 3 Impulse in der Sekunde) zu erfassen. Die Kosten für die beiden Versionen belaufen sich auf brutto 225,70 € beziehungsweise 285,20 € und sind damit verhältnismäßig preisgünstig (GÜTTICH, 2004).

Die ISOagriNET konforme Integration der Ethernetbox in das Stallnetz ist auf Hardwareebene problemlos möglich, da sie bereits über die notwendige Ethernetschnittstelle verfügt. Sie ist somit sehr gut dazu geeignet, die Werte von Sensoren oder Verbrauchsmessern im Netzwerk bereitzustellen.

Eine an einem Analogeingang anliegende Spannung oder der jeweilige Schaltzustand eines Digitaleinganges und der dazugehörige Wert des internen Schaltzustandzählers der Ethernetbox sind mithilfe des Transmission Control Protocol (TCP) abfragbar (vgl. Kapitel 5.6.3).

Für die ISOagriNET konforme Bereitstellung sind der Kommunikationsablauf und das Nachrichtenformat allerdings ungeeignet, da ISOagriNET eigene vorgibt. Kapitel 5.6.3 stellt die entwickelte Softwarekomponente namens Ethernetbox Service vor, die eine Umsetzung des Ethernetbox Protokolls in ADIS/ADED durchführt.

5.4.3 Management PC, Datenbank Server und virtueller Server

Der Betrieb der Farming Cell setzt mindestens einen vollwertigen Computer im Versuchsstall voraus. Um eine bessere Trennung zwischen der Datenhaltung und anderen Bereichen zu erzielen, verwendet die Farming Cell zwei gleichwertige Maschinen mit folgenden Eigenschaften.

- Betriebssystem: Windows XP (Management PC) bzw. Debian Lenny (Datenbank Server)
- Prozessor: Intel Core2Duo 6420 2,13 GHz
- Arbeitsspeicher: 3,11 GB
- Festplatte: 160 GB SATA
- 2 x PCI: beide frei

Die Tatsache, dass zum einen ein Windows und zum anderen ein Linux Betriebssystem zum Einsatz kommt, ermöglicht Test und Betrieb der entwickelten Softwarekomponenten in beiden Welten.

Um den Aspekt der Datensicherheit Rechnung zu tragen, wird im Rechenzentrum der Universität Hohenheim ein **virtueller Server** betrieben. Dieser ist mit dem Betriebssystem Windows Server 2008 ausgestattet und bietet eine vollständige Entwicklungsumgebung für die Farming Cell. Sicherungen der Farming Cell Datenbank sowie aller entwickelter Programme finden sich ebendort. Dem virtuellen Server ist die durch das Rechenzentrum der Universität Hohenheim registrierte URL farmingcell.de zugeordnet. Geeignete Applikationen für die Steuerung des Servers werden durch das Rechenzentrum der Universität Hohenheim bereitgestellt (vgl. BIRGELS, 2009).

Die folgende Tabelle gibt einen Überblick über die auf den drei Maschinen betriebenen Dienste.

Tabelle 5.9: Softwaredienste einzelner Maschinen

Software	Management PC	Datenbank Server	Virtueller Server
Online für Windows (Schauer)	**x**	-	x
Klima 3.3.7 (Möller)	**x**	-	x
Lumasoft Gas und Microsoft SQL Server x2005 (Lumasense)	**x**	-	x
Link3000 (Tru-Test)	**x**	-	x
JetPort Commander	**x**	-	x
VNC Server	**x**	**x**	x
VisualSVN Server	x	x	**x**
Tomcat Webserver	x	x	**x**
XAMPP	**x**	x	x
JDK 6.x	**x**	**x**	**x**
Eclipse J2SE	x	x	x
Eclipse J2EE	x	x	x
Eclipse BIRT	x	x	x
Schauer Service	**x**	x	x
TruTest Service	**x**	x	x
Ethernetbox Service	**x**	x	x
ISOagriNET Parser	x	**x**	x
Datenbank	x	**x**	**x**
Info Mailer	x	**x**	x

Webapplikation	x	x	**x**
Reporting Applikation	x	x	**x**
phpMyAdmin	**x**	x	x
REST Service	x	x	**x**

x – lauffähig
x – lauffähig und genutzte Variante
- – nicht lauffähig

Obgleich mit den genannten Maschinen Möglichkeiten bereitstehen, die Datenbank der Farming Cell zu sichern und mehrfach vorzuhalten, wurde eine **weitere Datenbank** in das Gesamtkonzept einbezogen. Diese wird durch das Rechenzentrum der Universität betrieben. Es handelt sich um eine MySQL Datenbank der Version 5.0.77, welche auf einer durch das genannte Rechenzentrum administrierten virtuellen Maschine (URL: mysqlb-vm.rz.uni-hohenheim.de) läuft. Der Beweggrund für die Verwendung dieser zusätzlichen Datenbank liegt einerseits in der besseren Performance der Hardware des genannten Rechenzentrums begründet. Die rechenaufwendigen Datenbankoperationen, wie sie beispielsweise die Reporting Applikation erfordern, machen den Einsatz leistungsfähiger Hardware notwendig. Desweiteren wird die Datensicherheit für im Rechenzentrum der Universität Hohenheim befindliche Daten besser eingeschätzt, als für diejenigen auf den oben genannten Maschinen. Eine **regelmäßige Sicherung** der Produktivdatenbank auf die Datenbank des Rechenzentrums der Universität Hohenheim wäre daher in Zukunft sinnvoll. Mögliche Ansätze werden in Kapitel 7.1.3 diskutiert.

5.5 Sensoren und Verbrauchsmesser

Die im Versuchsstall installierten Sensoren und Verbrauchsmesser sind mittels **Hohenheimer Messwerterfassung** oder **Ethernetbox** ISOagriNET konform in das Netzwerk der Farming Cell eingebunden (vgl. Kapitel 5.4.1 und 5.4.2). Die folgende Abbildung gibt eine Übersicht über Typ, Anzahl und Position der im Versuchsstall installierten Sensoren und Verbrauchsmesser. Ebenfalls stellt es die X und Y Achsen des Koordinatensystems dar. Die nicht eingezeichnete Z Achse ist die Lotachse. Die Räume Abteile 1 und 2 sowie Vorraum besitzen eigene Nullpunkte, welche sich in der Abbildung 5.8, jeweils in der unteren linken Ecke befinden.

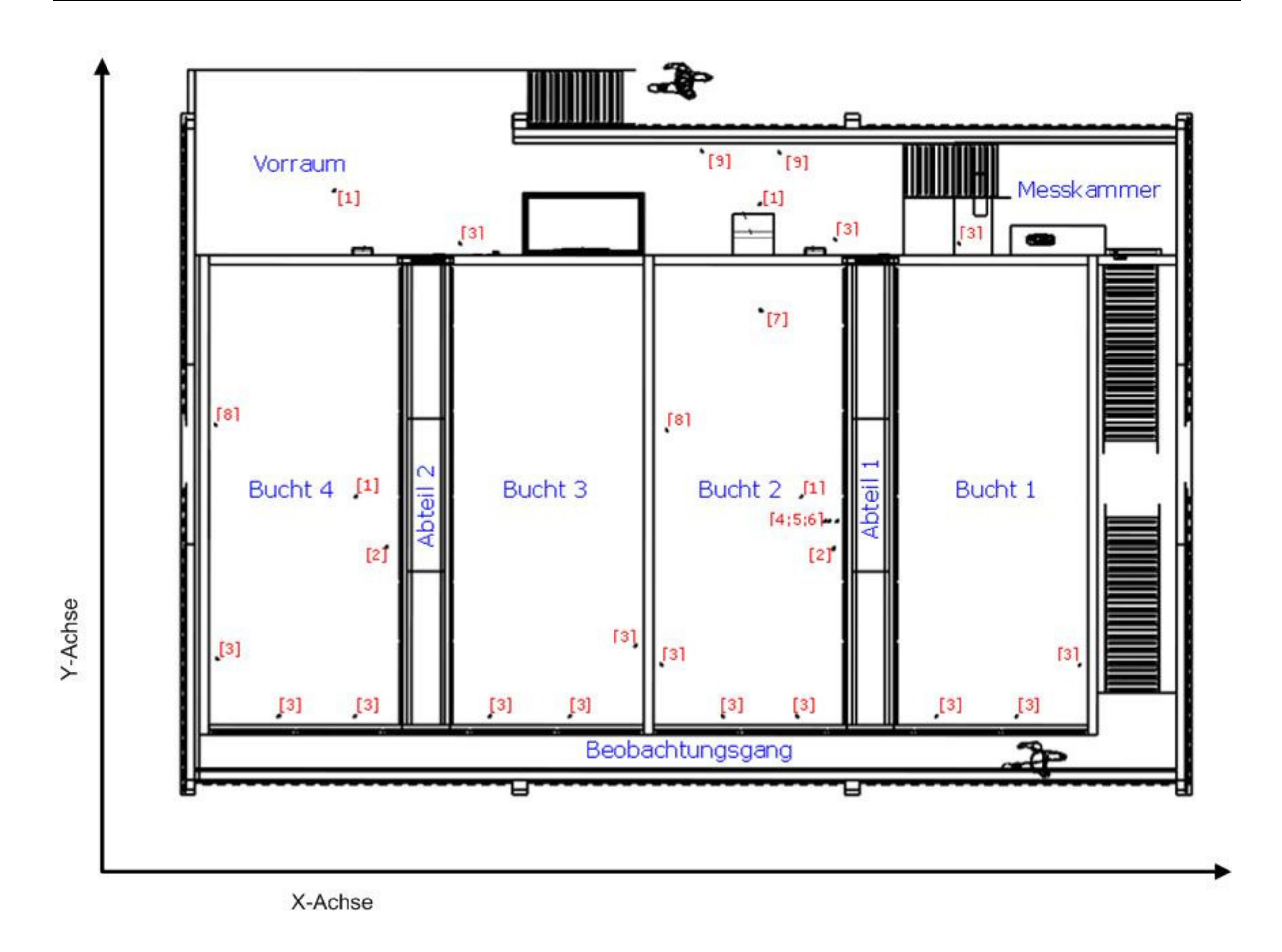


Abbildung 5.8: Sensoren im Versuchsstall

Legende zu Abbildung 5.8

Tabelle 5.10: Sensoren und Verbrauchsmesser im Versuchsstall

Sensor / Verbrauchsmesser	Anzahl	Signal
[1] Temperatur	4	Spannung
[2] Luftfeuchte	2	“
[3] Wasservolumenmesser	15	Impuls (S0)
[4] Helligkeit	1	Spannung
[5] NH_3	1	“
[6] CO_2	1	“
[7] Differenzdruck	1	“
[8] Wärmeenergiezähler	2	Impuls (S0)
[9] Stromenergiezähler	9	“

Für die Wasservolumenmesser der zwölf Tränken wurden am 25.02.2008 mittels einer definierten Wasserentnahme individuelle **Korrekturfaktoren** bestimmt. Diese fanden im Rahmen der Datenerhebung Anwendung und beeinflussen die in Kapitel 6 dargestellten Ergebnisse.

Tabelle 5.11: Korrekturfaktoren der Tränkenippelwassermesser

Abteil und Tränkenippel	Korrekturfaktor
Abteil1 Z1	0,840
Abteil1 Z2	0,747
Abteil1 Z3	0,764
Abteil1 Z4	0,793
Abteil1 Z5	0,751
Abteil1 Z6	0,736
Abteil2 Z1	0,790
Abteil2 Z2	0,923
Abteil2 Z3	0,803
Abteil2 Z4	0,872
Abteil2 Z5	0,789
Abteil2 Z6	0,873

Die ADED Entitäten Entity 1 und 2 (vgl. Kapitel 5.2.1) geben die an die HME und die Ethernetbox anschließbaren Sensor- und Verbrauchsmessertypen vor. Die Ethernetbox unterstützt alle in Tabelle 5.10 genannten Typen, die HME nur diejenigen, die als Signal einen Strom oder eine Spannung ausgeben.

5.6 Software

Dieses Kapitel befasst sich mit den Softwarekomponenten, die das Erfassen, Bereitstellen und Manipulieren sowie das Auswerten der Farming Cell Daten ermöglichen. Mit Ausnahme des nachfolgend vorgestellten Datenbanksystems und der dazu gehörenden Administrationsschnittstelle phpMyAdmin, handelt es sich hierbei um Eigenentwicklungen, denen gemein ist, dass sie in der Programmiersprache Java entwickelt wurden. Dies bietet den Vorteil der Plattformunabhängigkeit und somit Variationsmöglichkeiten hinsichtlich der beheimatenden Computer (vgl. Tabelle 5.9).

5.6.1 Schauer Service

Die im Versuchsstall verbaute Fütterungsanlage der Firma Schauer Agrotronic GmbH wurde mithilfe eines RS485 auf Ethernet Adapters an das lokale Netzwerk angebunden (vgl. Kapitel 5.3.1). Es sind, abgesehen von dem Verwaltungsprogramm Schauer Online, zwei Wege der **Kommunikation** möglich.

1. Verwendung einer durch die Firma Schauer bereitgestellten Dynamic Link Library (DLL) deren Funktionen aus Java mittel Java Native Interfaces (JNI) genutzt werden können.
2. Direkte Kommunikation aus Java bei eigener Implementierung der benötigten Funktionen samt des Schauer eigenen Verschlüsselungsalgorithmus.

Beide Varianten bergen Vor- und Nachteile, die die folgende Tabelle verdeutlicht.

Tabelle 5.12: Kommunikation mit dem Fütterungscomputer

Aspekt	mit DLL	ohne DLL
Entwicklungsaufwand		
- Universität	gering	hoch
- Firma Schauer	mittel	keiner
Entwicklungskosten (monetär)	gering	keine
Funktionale Erweiterungen	nur durch Firma Schauer möglich	universitätsseitig möglich
Fehlerwahrscheinlichkeit	sehr gering	mittel
Kommunikation zwischen Fütterungssteuerung und		
- HME	nicht möglich	möglich
- Computer	möglich	möglich
ISOagriNET Kompatibilität	nur sehr umständlich herzustellen	leicht herzustellen
Haftung	vorrangig Firma Schauer	vorrangig Universität

Zunächst wurden beide Wege der Implementierung verfolgt. Eine prototypische Implementierung der Kommunikation ohne DLL (Projekt Schauer_Fuetterung_direkte_Kommunikation) wurde aufgrund des enormen Entwicklungsaufwands und der mit der DLL verfügbaren Alternative verworfen. Auf das im genannten Projekt implementierte **Protokoll der Fütterung** soll an dieser Stelle nicht weiter eingegangen werden, stattdessen sei auf Erläuterungen zu eben diesem verwiesen (Ablauf_Kommunikation.txt im Projekt Schauer_Fuetterung_direkte_Kommunikation des VisualSVN Servers).

Zwar bedeutet die Verwendung der durch die Firma Schauer speziell entwickelten DLL eine Einschränkung auf einen vorgegebenen Satz Funktionen und den Verzicht auf ISOagriNET Kompatibilität. Dem entgegen steht jedoch, dass es sich bei der Fütterungsanlage um eine sensible Komponente handelt und die Verantwortung für implementierungsbedingte Fehlfunktionen bei Verwendung der DLL größtenteils beim Hersteller liegt.

Die **Dynamic Link Library** trägt den Namen SCHAPI (R1001).dll und bietet die in der folgenden Tabelle erläuterten sechs Funktionen.

Tabelle 5.13: Funktionen der SCHAPI DLL

Funktion	Erläuterung
int GetSCHAPIVersionInfo (int r1, int r2, int r3, int r4)	Liefert die Versionsnummer der SCHAPI DLL.
int CmdSCHAPIApplyCommunicationSettings (String comport, int baud, String parity, int timeout, int retries)	Konfiguriert den COM-Port am PC. Diese Funktion retourniert immer 1.
int CmdSCHAPIEstablishConnection (int computerno)	Stellt die Verbindung zum Fütterungscomputer her. Rückgabewerte: - 1: Verbindung erfolgreich hergestellt - 99: Fütterungscomputer antwortet nicht - 98: Prüfsumme des empfangenen Datenpaketes falsch

int GetSCHAPINoofValves (int computerno, int noofvalves)	Liefert die Anzahl Ventile der Fütterung.
int GetSCHAPIValveConsumption (int computerno, int valveno, int noofanimals, int quantityg, int componentno01, int componentno02, int componentno03, int componentno04, int componentno05, int componentno06, int componentno07, int componentno08, int componentno09, int componentno10, int componentno11, int componentno12, int componentno13, int componentno14, int componentno15, int componentno16)	Liefert die Ventil- und Komponentenverbräuche sowie die Tieranzahl für ein Ventil (1 bis noofvalves). Die Ventilverbräuche (quantityg) werden in der Steuerung nach jedem Abholen zurückgesetzt.
int SetSCHAPIValveData (int computerno, int valveno, int noofanimals, int weightg)	Setzt die Anzahl Tiere und das Tierdurchschnittsgewicht für ein Ventil.

Mithilfe der DLL wurde ein Programm Namens **Schauer Service** implementiert. Es arbeitet auf Kommandozeilenebene und bietet zwei Funktionalitäten, die zeitgesteuert ausgeführt werden, aber auch manuell angestoßen werden können.

1. Abrufen aller Ventilverbräuche sowie dazugehöriger Tieranzahl aus der Fütterungssteuerung und anschließendes Speichern in der Farming Cell Datenbank (Tabelle feed_consumption)
2. Überprüfung ob in der Farming Cell Datenbank tagesaktuelle Wiegedaten vorhanden sind (Tabelle entity_610011). Für Ventile/Buchten auf die dies zutrifft, werden die Informationen Tieranzahl und Tierdurchschnittsgewicht an die Fütterung weitergegeben. Die Mitglieder der Mailingliste (vgl. Kapitel 5.6.7) werden im Falle der Datenübermittlung an die Fütterung detailliert über den Vorgang informiert (vgl. Abbildung 5.10).

```
C:\WINDOWS\system32\cmd.exe
C:\Dokumente und Einstellungen\ITFT\Desktop\ADIS Schauer Client Server>java -jar
 "ADIS Schauer PC Client Server.jar"

Starting Schauer Client/Server.
Type 'h' for help.
Connected with feeding: true
Number of valves: 4
h

*************** Help ***************
The following commands are valid:
e - Get valve consumption for all 4 valves and store it in DB.
f - Check if settings in feeding need to be updated.
q - Quit application.
h - Prints this help.
Asking Schauer for valve consumption: Thu Sep 24 22:30:52 CEST 2009
24.09.2009 22:31:03
-Todays feeding data: <valveConsumption=2079000, numberOfAnimals=28, penNumber=1
>
24.09.2009 22:31:03
-Todays feeding data: <valveConsumption=2052000, numberOfAnimals=29, penNumber=2
>
24.09.2009 22:31:03
-Todays feeding data: <valveConsumption=662000, numberOfAnimals=9, penNumber=3>
24.09.2009 22:31:03
-Todays feeding data: <valveConsumption=838000, numberOfAnimals=13, penNumber=4>

Checking pen 1.
Values in feeding had to be updated.

Scale says there are 26 animals in pen 1 , DB says 28.
The number of animals from DB will be used.

The following values have been set for pen number 1.
Number of animals: 28
Average weight in g: 109692


Checking pen 2.
Values in feeding had to be updated.

Scale says there are 26 animals in pen 2 , DB says 29.
The number of animals from DB will be used.

The following values have been set for pen number 2.
Number of animals: 29
Average weight in g: 99846


Checking pen 3.
No changes need to be done!
Checking pen 4.
No changes need to be done!
```

Abbildung 5.9: Schauer Service

Abbildung 5.9 zeigt die Software Schauer Service im Betrieb. Ihre **Konfiguration** erfolgt zum einen mithilfe der Datei setup.properties, welche die Verbindungsparameter für die serielle Schnittstelle (comport, baudRate, parity, timeout, retries), die Nummer des Fütterungscomputers (computerNo) und den Zeitpunkt für die automatische Ausführung (run_Hour, run_Minute) festlegt. Zum anderen werden in der Datei MySQL_DB.properties die für die Verbindungsaufnahme mir der Farming Cell Datenbank erforderlichen Parameter gesetzt. Nachfolgend finden sich Beispielkonfigurationen.

Inhalt der Datei setup.properties

```
computerNo = 1
comport = COM3
baudRate = 9600
parity = EVENPARITY
timeout = 1000
retries = 3
run_Hour = 23
run_Minute = 55
```

Inhalt der Datei MySQL_DB.properties

```
driver = com.mysql.jdbc.Driver
DB_SERVER = ***.***.***.***
DB_NAME = farmingcell
password = ********
user = farmingcell
url = jdbc:mysql://
```

Ist eine Aktualisierung der Tieranzahl und des Tierdurchschnittsgewichtes aufgrund tagesaktueller Wiegedaten durchgeführt worden, wird eine **E-Mail** an die Mailingliste verschickt, wie sie in der folgenden Abbildung beispielhaft zu sehen ist.

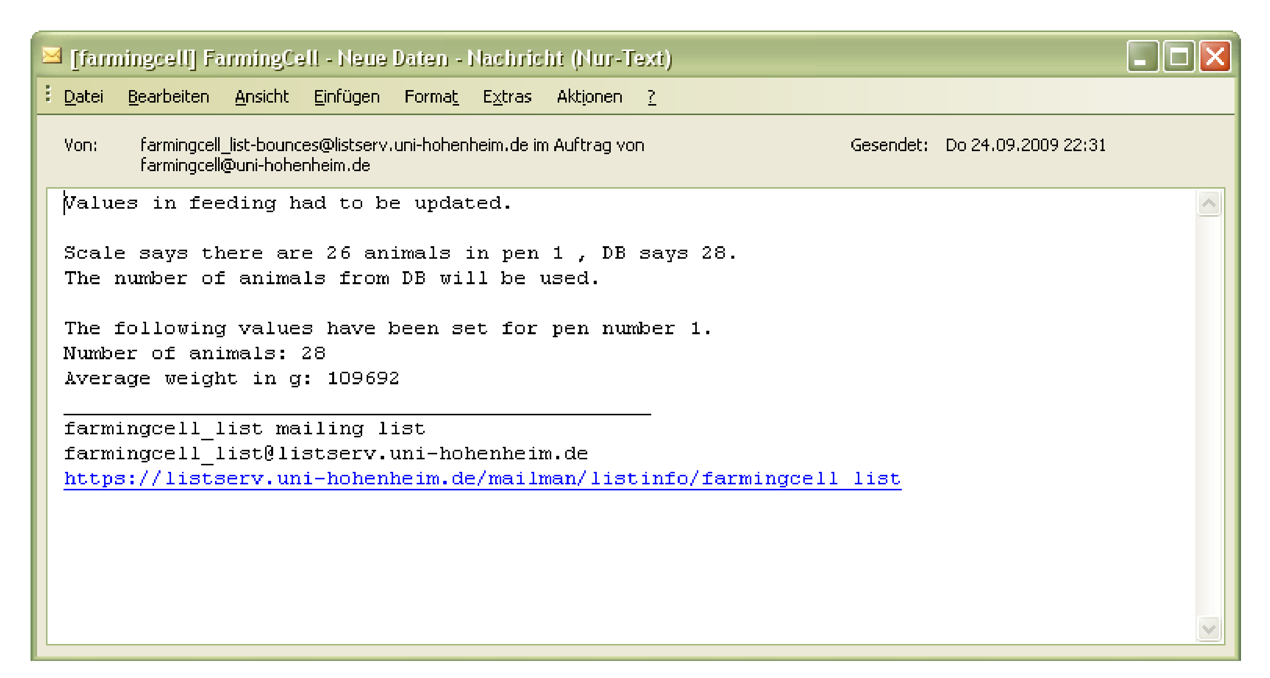


Abbildung 5.10: E-Mail bei Fütterungsanpassung

Zu beachten ist, dass es zu einer Uneindeutigkeit hinsichtlich der aktuellen Tieranzahl kommen kann. Dies ist der Fall, sollten die aus den Datenbanktabellen entity_610011 (Wiegedaten) und animal (Tierbestand) ermittelten **Tieranzahlen** einer Bucht nicht identisch sein. Dies kann zwei Gründe haben:

1. Während des Wiegevorganges wurden nicht alle Tiere erfasst (Doppelterfassungen sind nicht schädlich) → zu wenig Tiere in Tabelle entity_610011.
2. Der Tierbestand wurde nicht (korrekt) gepflegt → zu viele oder zu wenig aktive Tiere in Tabelle animals.

Sollte eine **Diskrepanz** auftreten, wird die Tieranzahl in der Fütterung entsprechend der Tabelle animal gesetzt. Das Auslesen der aktuell in der Fütterung gesetzten Tieranzahl ist nicht ohne Weiteres möglich und somit keine Option, um einer Diskrepanz zu begegnen. Eine solche Abfrage hätte das Rücksetzen des Ventilverbrauches zur Folge und wäre an anderer Stelle von Nachteil (vgl. Funktion GetSCHAPIValveConsumption in Tabelle 5.13). Das zu setzende Tierdurchschnittsgewicht wird auf Basis der Daten der Datenbanktabelle entity_610011 (vgl. Anhang A) errechnet.

5.6.2 TruTest Services

Das Terminal XR3000 der im Versuchsstall installierten Tierwaage der Firma Tru-Test verfügt über zwei Eingänge zum Anschluss der Wiegeplattform und des RFID Lesegerätes. Ferner besitzt es zwei serielle Schnittstellen, welche unter Verwendung des Tru-Test eigenen aber offengelegten Protokolls, die Kommunikation mit dem Waagenterminal ermöglichen (vgl. Anonymus, 2004a). Auf der Gegenseite ist die HME angeschlossen, welche ebenfalls über eine serielle Schnittstelle verfügt.

Das Ziel, die Einzeltiergewichte einer Wiegung automatisch in die Datenbank zu übertragen, wird durch die zuvor genannte Art des Anschlusses sowie die zwei nachfolgend vorgestellten Softwarekomponenten erreicht. Die Kommunikation, d.h. das Anfragen neuer Wiegedaten und deren anschließende Übermittlung bis hin zur Ablage in der Datenbank, lässt sich hinsichtlich des Kommunikationsmediums in zwei Abschnitte unterteilen. Zum einen erfolgt in Abschnitt eins eine Kommunikation zwischen Waagensteuerung und HME auf Basis der seriellen **Schnittstelle**. Im zweiten Abschnitt kommuniziert die HME entsprechend ISOagriNET ethernetbasiert mit einem Computer im Netzwerk.

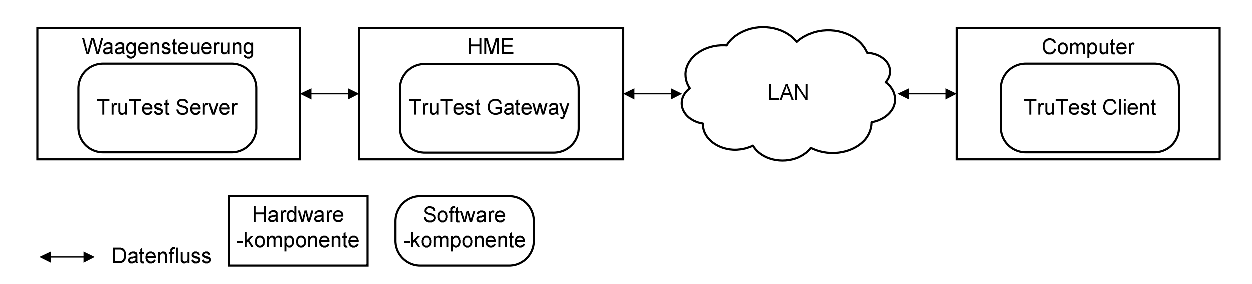


Abbildung 5.11: Anbindung der Tierwaage

Die HME fungiert somit als Gateway zwischen Waagensteuerung und Stallnetz. Sie muss neben zwei Hardwareschnittstellen daher auch zwei **Kommunikations- und Datenprotokolle**, die der Firma Tru-Test und ISOagriNET, unterstützen. Die zu diesem Zwecke für die HME entwickelte Software wird im Folgenden als TruTest Gateway bezeichnet. Die in der Waagensteuerung und auf dem Computer befindlichen Kommunikationsendpunkte tragen die Namen TruTest Server sowie TruTest Client (vgl. Abbildung 5.12).

Der **Ablauf einer Kommunikation** setzt sich zusammen aus der Anfrage durch den TruTest Client an das TruTest Gateway, welches wiederum eine inhaltlich identische, syntaktisch jedoch abweichende Anfrage an den TruTest Server sendet. Dessen Antwort verarbeitet das Gateway und reicht im Anschluss eine inhaltlich identische und wiederum syntaktisch abweichende Nachricht zurück an den TruTest Client. Dieser verarbeitet die Antwort und schreibt die gewonnen Daten in die Datenbank.

An dieser Stelle werden lediglich die Details des zweiten Kommunikationsabschnitts zwischen HME und Computer beispielhaft erläutert, da die Kommunikation in Abschnitt eins durch die von der Firma Tru-Test bereitgestellte Dokumentation erläutert wird (vgl. Anonymus, 2004a).

Abschnitt zwei umfasst zwei Kommunikationsschritte. Zuerst erfolgt die Anfrage des TruTest Clients nach tagesaktuellen Wiegedaten an das TruTest Gateway im Format ADIS/ADED. Diese sieht beispielsweise wie folgt aus.

```
DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020091014140348Management_PC
SN610011006100760802009101420091014
RN610011006101761500061007608000610035041
TN
```

Die mit DH990054 und VH990054 beginnenden Zeilen enthalten den Standard ISOagriNET Header. Die Abkürzungen DH und VH stehen für Definition Header und Value Header. Die Zahl 990054 ist die Nummer der Entität Namens isoagrinet_header.
Die letzte Zeile mit dem Inhalt TN schließt die Nachricht ab.
In den Zeilen SN610011 und RN610011 (Entity 610011 WIEGEN) ist die eigentliche Anfrage enthalten. Eine R (Request) Zeile definiert dabei, welche Informationen als Antwort gewünscht sind. S (Search) Zeilen enthalten Bedingungen, die eine Einschränkung der Ergebnismenge erlauben. Es ist zulässig, mehrere Search Zeilen zu verwenden, jedoch darf nur eine R Zeile existieren (vgl. ISO, 2009).

Eine **Request** Zeile ist folgendermaßen aufgebaut:

RN<Entity_Nummer><Item_1_Nummer><Item_1_Länge><Item_1_Auflösung>…<Item_n_Nummer><Item_n_Länge><Item_n_Auflösung>

Im vorliegenden Beispiel werden drei Items einer Entität angefragt (Leerzeichen im nachträglich eingefügt).

```
RN 610011 00610176150 00610076080 00610035041
```

610011	- Entität 610011 (WIEGEN)
00610176 150	- Item 610176 (tier_id); Länge 15, Auflösung 0
00610076 080	- Item 610076 (WIEGEDATUM); Länge 8, Auflösung 0
00610035 041	- Item 610035 (GEWICHT); Länge 4, Auflösung 1

Die **Search** Zeile ist folgendermaßen aufgebaut.

SN<Entity_Nummer><Item_Nummer><Item_Untergrenze><Item_Obergrenze>

Die Entity Nummer muss der der RN Zeile entsprechen. Im folgenden Beispiel ist genau eine Bedingung formuliert.

```
SN 610011 00610076080 20091014 20091014
```

610011	- Entität 610011 (WIEGEN)
00610076 080	- Item 610076 (WIEGEDATUM); Länge 8, Auflösung 0
20091014	- Untergrenze für das Item 610076; der 14.10.2009
20091014	- Obergrenze für das Item 610076; der 14.10.2009

Die vorgestellte Anfrage lässt sich verbal wie folgt formulieren:
Liefere die Parameter tier_id, WIEGEDATUM und GEWICHT aller am 14.10.2009 stattgefundenen Wiegeereignisse zurück.

Die **Antwort** des TruTest Gateways auf der HME sieht auszugsweise wie folgt aus:

```
DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090512113059HME
DN610011006101761500061007608000610035041
VN610011969000000367000200910140873
VN610011969000000367007200910140845
VN610011969000000366973200910140921
...
TN
```

Die Zeilen DH990054 und VH990054 enthalten wiederum den Standard ISOagriNET Header (Entity 990054 isoagrinet_header) und auch diese Nachricht wird durch eine TN Zeile abgeschlossen.
Die Zeilen VN und DN enthalten die eigentlichen Nutzdaten sowie deren Definition (ISO, 2009). Die Zeile DN (Definition Normal Data) beschreibt den Inhalt und die Struktur der folgenden VN (Value Normal Data) Zeilen. Es sind beliebig viele VN Zeilen zulässig.

```
DN 610011 00610176150 00610076080 00610035041
```

Die DN Zeile ist inhaltlich identisch mit der bereits vorgestellten RN Zeile und wird daher nicht erläutert.

Der Aufbau einer **Nutzdatenzeile** (VN) gestaltet sich wie folgt:

```
VN 610011 969000000367000 20091014 0873
```

610011	- Entität 610011 (WIEGEN)
969000000367000	- Wert für Item 610176 (tier_id), 969000000367000
20091014	- Wert für Item 610076 (WIEGEDATUM); 14.10.2009
0873	- Wert für Item 610035 (GEWICHT); 87,3 kg

Die Antwortnachricht wird auf Seiten des TruTest Clients abgearbeitet und die Daten in die Datenbank geschrieben (Tabelle entity_610011).

Zu beachten ist, dass für die Kommunikation **TCP** genutzt wird. Der Grund, in diesem Fall nicht UDP Multicast zu verwenden, liegt darin begründet, dass zum einen die Länge der Nachricht nicht vorhersagbar ist und zum anderen die vollständige und fehlerfreie Übertragung sichergestellt sein muss.

Der **Ausführungsort** des TruTest Gateways ist die an die Waagensteuerung angeschlossene **HME**. Die Software wird automatisch bei HME Start geladen und ist permanent erreichbar. Eine grafische Oberfläche existiert nicht.

Der aus einer ausführbaren Java Archiv (JAR) Datei und Konfigurationsdateien (siehe unten) bestehende TruTest Client kann auf einem beliebigen **Computer** im Netzwerk ausgeführt werden. Voraussetzungen für dessen Ausführung sind neben einer Java Runtime die Erreichbarkeit des Datenbank Servers und der HME sowie das Vorhandensein des Agricultural Data Exchange Dictionaries (ADED) im Format SQLite in der Version 1.02 wie es die Software Tisan (vgl. Kapitel 4.2) verwendet.
Der TruTest Client besitzt drei Konfigurationsdateien.

1. setup.properties

   Diese Datei enthält die IP Adresse der HME und die volle Stunde, zu der die Wiegedaten täglich abgefragt werden sollen.

   ```
   tini_ip=***.***.***.***
   run_Hour=23
   ```

2. MySQL_DB.properties

   In dieser Datei sind die für den Aufbau einer Verbindung zum Datenbankserver benötigten Parameter abgelegt.

   ```
   driver = com.mysql.jdbc.Driver
   DB_SERVER = ***.***.***.***
   DB_NAME = farmingcell
   password = ********
   user = farmingcell
   url = jdbc:mysql://
   ```

3. SQLite_DB.properties

Die für den Verbindungsaufbau zu der dateibasierten ISOagriNET Data Dictionary SQLite Datenbank erforderlichen Parameter sind in dieser Datei hinterlegt.

```
driver = org.sqlite.JDBC
file = C:/ISOagrinet/TISaN_1_02/adis_dd.db
url = jdbc:sqlite:
```

Der TruTest Client ist auf **Kommandozeilenebene** steuerbar (vgl. Abbildung 5.12). Die möglichen Optionen werden bei Programmstart angezeigt. Zusätzlich zur täglichen, automatischen Ausführung ist es möglich, die Abfrage der Wiegedaten manuell anzustoßen (Option e). Den Übertragungsprozess betreffende Statusmeldungen werden im Kommandozeilenfenster ausgegeben.

```
C:\WINDOWS\system32\cmd.exe
C:\Dokumente und Einstellungen\ITFT\Desktop\adis_trutest>java -jar "ADIS_TruTest
_Client_PC.jar"

Starting TruTest Client.
Type 'h' for help.
14.10.2009 22:00:30 ClientSocket <init>
FEIN: Connected to ADIS DB!
14.10.2009 22:00:36 ClientSocket <init>
FEIN: Connected to measurement DB!
h

********** Help **********
The following commands are valid:
e - Get data from TruTest now.
i - Prints some infos.
q - Quit application.
h - Prints this help.

i
The automatic request is running at about 23:00 (+/- one minute).
The TINI connected with TruTest should be reachable under IP 144.41.72.181.
```

Abbildung 5.12: TruTest Client

5.6.3 Ethernetbox Service

Als eine Möglichkeit der Messwerterfassung und -bereitstellung kommen die in Kapitel 5.4.2 vorgestellten Ethernetboxen zum Einsatz. Die Geräte besitzen neben einer grafischen Benutzerschnittstelle in Form eines Webfrontends eine weitere auf dem Transmission Control Protocol (TCP) basierende. Das verwendete Nachrichtenformat zeichnet sich durch seine einfache Struktur aus (vgl. Tabelle 5.14) und ist in jeder Programmiersprache nutzbar. Über eine TCP Verbindung werden textuelle Steuerbefehle oder Anfragen an die Ethernetbox geschickt. Diese antwortet ebenso textbasiert. Die Ethernetbox unterstützt je nach Ausführung bis zu vier solcher Befehlstypen.

Tabelle 5.14: Ethernetbox Nachrichtenformat

Befehle	Antwort	Beschreibung	Ethernet-box Typ	
			1	**2**
pcmeasure.com1.1 (= Port 1) pcmeasure.com1.2 (= Port 2) pcmeasure.com1.3 (= Port 3) pcmeasure.com1.4 (= Port 4) pcmeasure.com2.1 (= Port 5) pcmeasure.com2.2 (= Port 6) pcmeasure.com2.3 (= Port 7) pcmeasure.com2.4 (= Port 8) pcmeasure.com3.1 (= Port 9) pcmeasure.com3.2 (=Port 10) pcmeasure.com3.3 (=Port 11) pcmeasure.com3.4 (=Port 12)	valid=1;value=100.0; oder valid=1;value= 0.0;	Zustände (high / low) der 12 Digitaleingänge lesen	ja	ja
counter.read.01; counter.read.02; . . . counter.read.12;	counter1;value=364 counter2;value=9572 . . . counter12;value=233	Zählerstände der zu den 12 Digitaleingängen gehörenden Zähler auslesen	ja	ja
counter.reset.01;password	counter1;value=942	Zählerstände der	ja	ja

=private counter.reset.02;password =private . . . counter.reset.12;password =private	counter2;value=23 . . . counter3;value=886	zu den 12 Digitaleingängen gehörenden Zähler auslesen und auf 0 setzen.		
pcmeasure.lpt1.1 (= Kanal 1) pcmeasure.lpt1.2 (= Kanal 2) pcmeasure.lpt1.3 (= Kanal 3) pcmeasure.lpt1.4 (= Kanal 4) pcmeasure.lpt2.1 (= Kanal 5) pcmeasure.lpt2.2 (= Kanal 6) pcmeasure.lpt2.3 (= Kanal 7) pcmeasure.lpt2.4 (= Kanal 8)	valid=1;value= 1.2697; . . .	Die an den acht Analogein- gängen anliegenden Spannungen auslesen.	nein	ja

Die **Kommunikation** erfolgt mit dem Protokoll TCP über den Port 4000 (SCHULZE, 2010). Unter Nutzung des genannten Befehlssatzes ist die Software Ethernetbox Service implementiert worden. Sie ist in der Lage, die Werte beliebig vieler Ethernetboxen abzufragen und anschließend ISOagriNET konform im Netzwerk zu publizieren (UDP Multicast). Die Software muss auf einem **Computer** im lokalen Netzwerk laufen und wird auf Kommandozeilenebene ausgeführt. Die Konfiguration erfolgt dateibasiert. Für jede abzufragende Ethernetbox ist eine **Konfigurationsdatei** anzulegen. Entsprechende Vorlagen für die Ethernetbox Typen 1 und 2 sind im SVN Projektordner vorhanden.

Beispielkonfiguration einer Ethernetbox Typ 2:

Inhalt der Datei type2_00-90-c2-d2-24-64.properties

```
general=192.168.1.1;2 #EthernetBox 2 - Vorraum
#formula;interval;autopublish;type;location;conversionFactor;correctionFact
or;coordinate_x;coordinate_y;coordinate_z;offset
1=(x);10;1;heat_meter;0:3.1.0;1.0;1.0;0.12;7.38;1.32;0.0;
2=(1000*x);10;1;water_meter;0:3.2.0;1.0;1.0;4.75;0.06;1.93;0.0;
3=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
4=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
5=(x);10;1;heat_meter;0:3.2.0;1.0;1.0;0.12;7.36;1.25;0.0;
6=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
7=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
8=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
9=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
10=(1000*x);10;1;water_meter;0:3.1.0;1.0;1.0;12.36;0.08;1.90;0.0;
11=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
12=(x);1;0;electric_meter;0:3.0.0;1.0;1.0;0.0;0.0;0.0;0.0;
13=(1537.9*x+96.289/1194.5906065275188789254201464 92);1;1;air_speed;0:3.1.0
;1.0;1.0;0.0;0.0;0.0;0.0;
14=(1537.9*x+96.289/1194.590606527518878925420146492);1;1;air_speed;0:3.2.0
;1.0;1.0;0.0;0.0;0.0;0.0;
15=(x-4/0.06);1;1;temperature;0:3.2.0;1.0;1.0;0.0;0.0;0.0;0.0;
16=(0.0020185*x*x*x*x)+(0.0052268*x*x*x)-(0.0074338*x*x)+(0.12617*x)-
(0.000061675*x*x*x*x*x);1;1;air_speed;0:3.2.0;1.0;1.0;0.0;0.0;0.0;0.0;
17=(12.5*x-25);1;1;nh3;0:3.1.0;1.0;1.0;3.01;3.44;0.92;0.0;
18=(99.8*x+1*10);1;1;co2;0:3.1.0;1.0;1.0;3.14;3.44;0.92;0.0;
19=(20*x);1;1;pressure;0:3.1.0;1.0;1.0;1.82;7.02;2.33;0.0;
20=(0.45*x-0,6);1;1;air_speed;0:3.1.0;1.0;1.0;0.0;0.0;0.0;0.0;
```

Die abgebildete Beispielkonfiguration einer Ethernetbox Typ 2 unterscheidet sich insofern von der eines Typs 1, als dass sie mehr Einträge enthält. Die Einträge 13 bis 20, welche sich auf die Analogeingänge beziehen, sind bei einer Typ 1 Konfiguration nicht vorhanden. Bei der Benennung der Konfigurationsdateien ist darauf zu achten, dass diese eindeutig erfolgt. Der Name der Konfigurationsdatei wird, das vorangestellte type1_ bzw. type2_ ausgenommen, in den verschickten ADIS/ADED Nachrichten als Absenderkennung verwendet. Sinnvoll ist daher die Verwendung der MAC Adresse, da diese weltweit einmalig ist.

Der Aufbau der Konfigurationsdatei ist zeilenorientiert. Jede Zeile enthält die Zuweisung von Werten zu einem Parameter. Zulässige Parameter sind neben general die Ziffern 1 bis 12 (Typ 1) bzw. 20 (Typ 2). Erstgenanntem Parameter werden, getrennt durch ein Semikolon, die IP Adresse der Ethernetbox und die Typangabe (1 oder 2) zugewiesen.

```
general=192.168.1.1;2
```

Dem schließen sich die numerischen Parameter an.

Jeder Konfigurationsdatei ist eine Kommentarzeile vorangestellt, die die zwingend einzuhaltende **Zuweisungsreihenfolge** für die numerischen Parameter beschreibt.

```
#formula;interval;autopublish;type;location;conversionFactor;correctionFact
or;coordinate_x;coordinate_y;coordinate_z;offset
```

Tabelle 5.15: Konfigurationsparameter des Ethernetbox Services

Zuweisungsparameter	Beschreibung	zulässige Werte
formula	nur für Analogeingänge zu verwenden; Formel zur Umrechnung der anliegenden Spannung in die Zielgröße	(x) für Digitaleingänge; für Analogeingänge jeder numerische Term; jeder Rechenschritt ist einzuklammern
interval	Abfrage und Publikationsinterval	1, 5, 10, 15, 20, 30, 60, 120, 180, 240, 360, 480, 720, 1440
autopublish	Indikator ob Messwertabfrage und Publikation erfolgen soll	0 (inaktiv), 1 (aktiv)
type	Typ des Sensors oder des Verbrauchsmessers	entsprechend der Entitäten 1 und 2; s.u.
location	Lokation nach ADIS/ADED; Formatierung:Betrieb:Stall.Ab teil.Bucht; lokaler Betrieb = 0	alphanumerische Werte
conversionFactor	Umrechnungsfaktor; wird mit Zählerwert oder anliegender Spannung multipliziert	nummerische Werte
correctionFactor	Korrekturfaktor; wird mit Zählerwert oder anliegender Spannung multipliziert	numerische Werte
coordinate_x, coordinate_y, coordinate_z	Koordinaten des Sensors oder Verbrauchsmessers	numerische Werte
offset (in aktueller Software Version nicht unterstützt)	nur für Digitaleingänge zu verwenden; Korrekturwert; wird zu einem Zählerwert hinzuaddiert	numerische Werte

Die durch den Ethernetbox Service verschickten Nachrichten basieren auf den Entitäten 1 (environment_values) und 2 (meter_values). Eine detaillierte Erläuterung der beiden Entitäten und deren Items ist in Kapitel 5.2.1 erfolgt.

Entsprechend den Entitäten 1 (environment_values) und 2 (meter_values) sind in den Konfigurationsdateien die in der folgenden Tabelle genannten Sensor- und Verbrauchsmessertypen zulässig (Benennung entsprechend ADIS/ADED Data Dictionary).

Tabelle 5.16: Sensor- und Verbrauchsmessertypen nach ADED

Sensoren	Verbrauchsmesser
temperature	electric_meter
humidity	water_meter
co2	heat_meter
nh3	gas_meter
ch4	
n2o	
unterdruck	
helligkeit	
air_speed	

Zur Ermittlung der Koordinaten ist ein einheitliches Koordinatensystem zu verwenden. Für den Versuchsstall auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof wurde ein Raster entwickelt, welches in Kapitel 5.5 erläutert wird.

Alle in Tabelle 5.15 genannten Parameter müssen gesetzt sein, wobei die Reihenfolge zwingend einzuhalten ist. Nachfolgend wird der **Zeilenaufbau** anhand eines Beispiels erläutert.

```
2=(1000*x);10;1;water_meter;0:3.2.0;1.0;1.0;4.75;0.06;1.93;0.0;
```

Es handelt sich hierbei um die Konfiguration des zweiten digitalen Einganges (2=). Die für die Berechnung des Messwertes zu verwendende Formel lautet (1000*x)[22], die Abfrage der anliegenden Spannung und die Publikation des errechneten Messwertes erfolgen alle 10 Minuten (10), Abfrage und Publikation sind aktiv (1), angeschlossen ist ein Wasservolumenmesser oder -zähler (water_meter), die Lokation gemäß ADIS/ADED ist der lokale Betrieb, Stall 3, Abteil 2 (0:3.2.0) (vgl. Tabelle 5.23), Umrechnungs- sowie Korrekturfaktor sind 1 (1.0;1.0), die Koordinaten x, y, z lauten 4.75, 0.06 und 1.93 (4.75;0.06;1.93), der Korrekturwert beträgt 0 (0.0);

Wird der Ethernetbox Service gestartet, werden die als aktiv (autopublish = 1) gekennzeichneten Einträge aller vorhandenen Konfigurationsdateien geladen und aufgelistet. Ebenso wird der Benutzer durch kurze Statusmeldungen über die Abarbeitung der Arbeitsschritte (Jobs) informiert. Abbildung 5.13 veranschaulicht dies anhand eines Beispiels.

[22] Einzelne Rechenschritte einer Formel sind einzuklammern, doppelte Klammern sind nicht zulässig. Punkt- vor Strichrechnung wird nicht berücksichtigt. Eine Exponentendarstellung existiert nicht, stattdessen ist die Basis mit sich selber zu multiplizieren.

```
C:\WINDOWS\system32\cmd.exe
C:\Dokumente und Einstellungen\ITFT\Desktop\ADIS_Ethernetbox>java -jar "AIDS Eth
ernetbox.jar"
adding job: Mac=00-90-c2-d2-26-af - Port=1 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=2 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=3 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=4 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=5 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=6 - Interval=1
adding job: Mac=00-90-c2-d2-26-af - Port=7 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=9 - Interval=1
adding job: Mac=00-90-c2-d2-26-af - Port=10 - Interval=1
adding job: Mac=00-90-c2-d2-26-af - Port=11 - Interval=10
adding job: Mac=00-90-c2-d2-26-af - Port=12 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=1 - Interval=1
adding job: Mac=00-90-c2-d3-47-0e - Port=2 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=3 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=4 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=5 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=6 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=7 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=8 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=9 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=10 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=11 - Interval=10
adding job: Mac=00-90-c2-d3-47-0e - Port=12 - Interval=10
adding job: Mac=00-90-c2-d2-24-64 - Port=1 - Interval=10
adding job: Mac=00-90-c2-d2-24-64 - Port=2 - Interval=10
adding job: Mac=00-90-c2-d2-24-64 - Port=5 - Interval=10
adding job: Mac=00-90-c2-d2-24-64 - Port=10 - Interval=10
adding job: Mac=00-90-c2-d2-24-64 - Port=13 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=14 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=15 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=16 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=17 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=18 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=19 - Interval=1
adding job: Mac=00-90-c2-d2-24-64 - Port=20 - Interval=1
running job: Mac=00-90-c2-d2-26-af - Port=6 - Interval=1 - Type=electric_meter
running job: Mac=00-90-c2-d2-26-af - Port=9 - Interval=1 - Type=electric_meter
running job: Mac=00-90-c2-d2-26-af - Port=10 - Interval=1 - Type=water_meter
running job: Mac=00-90-c2-d3-47-0e - Port=1 - Interval=1 - Type=water_meter
running job: Mac=00-90-c2-d2-24-64 - Port=13 - Interval=1 - Type=air_speed
running job: Mac=00-90-c2-d2-24-64 - Port=14 - Interval=1 - Type=air_speed
running job: Mac=00-90-c2-d2-24-64 - Port=15 - Interval=1 - Type=temperature
running job: Mac=00-90-c2-d2-24-64 - Port=16 - Interval=1 - Type=air_speed
running job: Mac=00-90-c2-d2-24-64 - Port=17 - Interval=1 - Type=nh3
```

Abbildung 5.13: Ethernetbox Service

Den in den Konfigurationsdateien hinterlegten Angaben entsprechend, werden die Messwerte ISOagriNET konform als **Multicast Nachricht** verschickt. Der Inhalt der Nachrichten besteht, je nachdem ob es sich um Messwerte eines Sensors oder eines Verbrauchsmessers handelt, aus einer Entität 1 (environment_values) oder 2 (meter_values). Das folgende Beispiel zeigt eine Entität 2 Nachricht.

```
DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090721175808Ebox:00-90-c2-d3-47-0e
DN000002009010024000090101317000000001042000000002042000000003042000000007082
VN0000020:3.2.4
20090721175808000011000130030000000000
ZN
```

Das folgende Kapitel 5.6.4 geht anhand einer Entität 1 Nachricht auf den Aufbau derartiger ADIS/ADED Nachrichten ein.

5.6.4 HME Software

Das Herzstück der in Kapitel 5.4.1 vorgestellten Hohenheimer Messwerterfassung (HME) ist ihre Software. Sie besteht aus den zwei Bausteinen **Logik** und **Webinterface.** Beide wurden in Java implementiert und speziell kompiliert, um auf dem Mikrocontroller der HME lauffähig zu sein (vgl. Anonymus, 2004b).
Die Logikkomponente übernimmt die Verwaltung der angeschlossenen One-Wire Module, die Errechnung der Messwerte sowie deren Publikation. Das Webinterface dient dem Benutzer als Konfigurationsschnittstelle und gibt darüber hinaus Auskunft über die zuletzt ermittelten Messwerte.

Die folgenden zwei Abbildungen vermitteln einen Eindruck der Benutzerschnittstelle. Eine umfassende Dokumentation der HME ist separat verfügbar (KUHLMANN u. RÖSSLER, 2009).

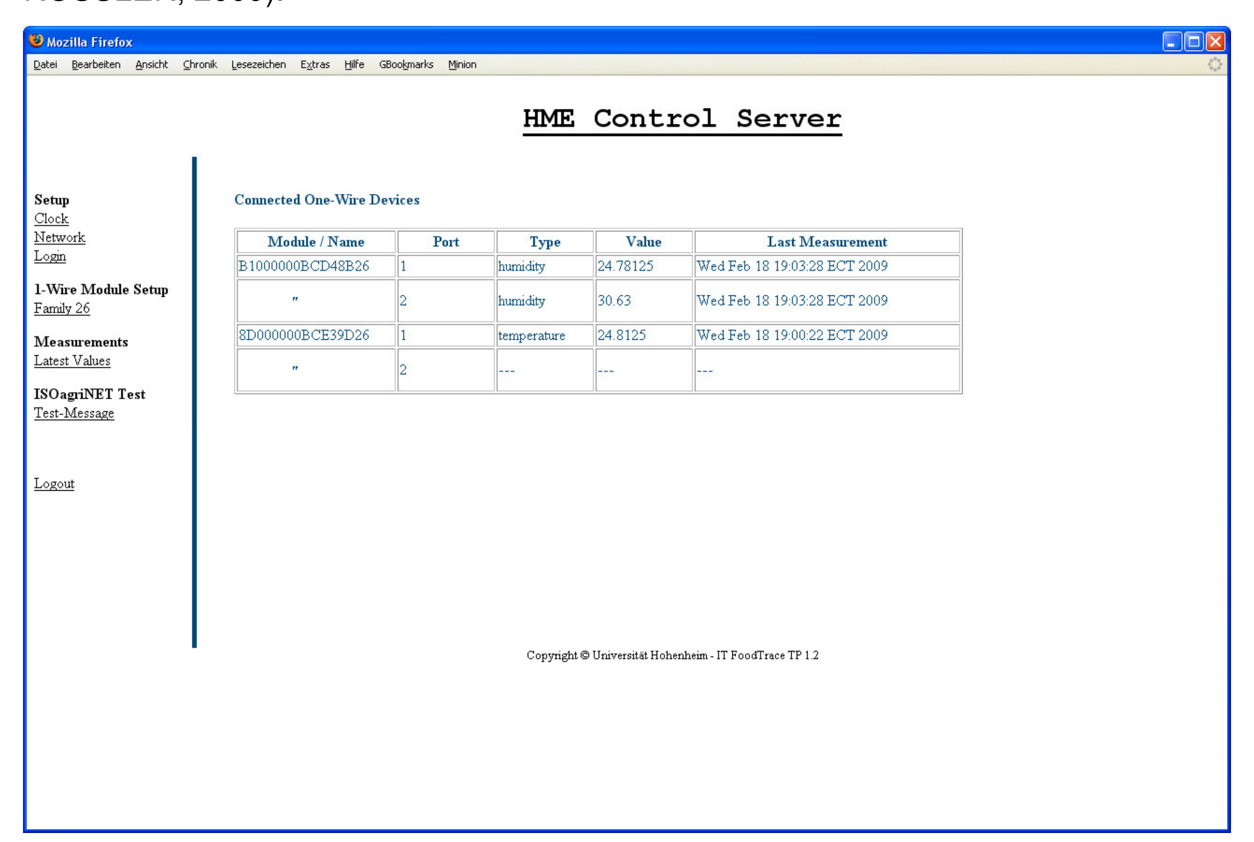


Abbildung 5.14: HME – Übersichtsseite

Abbildung 5.14 zeigt die Hauptseite des Webinterfaces, welche Auskunft über die angeschlossenen One-Wire Module und die Messwerte angeschlossener Sensoren gibt. In der folgenden Abbildung sind die Konfigurationsmöglichkeiten eines One-Wire Moduls dargestellt.

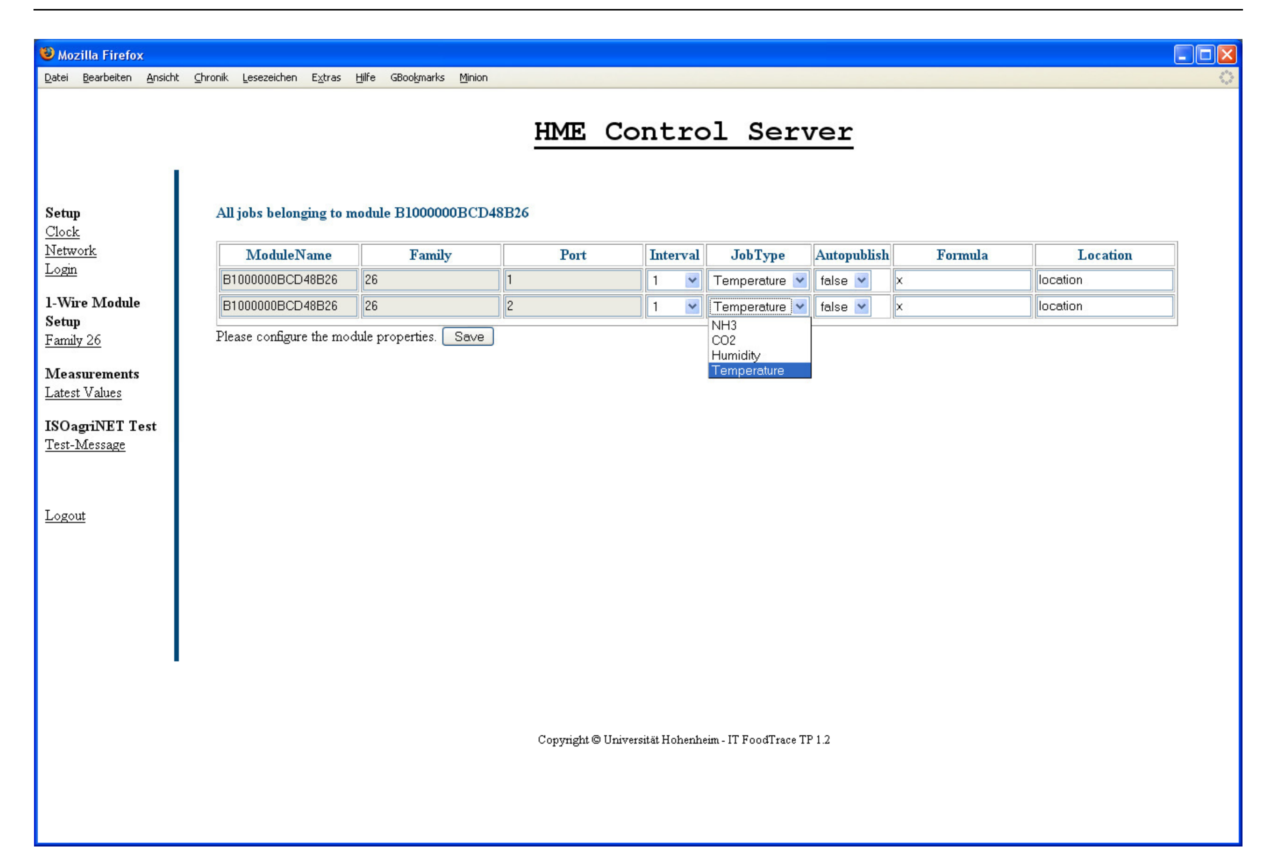


Abbildung 5.15: HME – Konfigurationsseite

Ist die **Konfiguration eines Moduls** (vgl. Tabelle 5.17) erfolgt, ist diese immer, bei angeschlossenem und bei nicht angeschlossenem Modul, verfügbar, da sie persistent auf der HME hinterlegt wird. Das Entfernen und wieder Anschließen eines Moduls hat keinen Verlust der Konfiguration zur Folge.

Tabelle 5.17: Einstellbare Parameter der One-Wire Module

Parameter	Bedeutung
Interval	Publikationsintervall in Minuten
JobType	Art des angeschlossenen Sensors
Autopublish	Im Netzwerk publizieren: ja / nein
Formula	Umrechnungsformel
Location	Installationsort des Sensors

Entsprechend der vorgenommenen Konfiguration werden die Messwerte ISOagriNET konform im Format ADIS/ADED publiziert. Hierfür findet die Entität 101000 Klima_Luftdaten Verwendung. Verbrauchsmesser sind daher in der vorgestellten Softwareversion, bei welcher es sich um die durch die DLG zertifizierte handelt, nicht auswählbar.

Aus eben diesem Grund existiert eine alternative, nicht durch die DLG als ISOagriNET konform zertifizierte, **Softwareversion** für die HME, die die bereits vorgestellten selbstdefinierten Entitäten 1 und 2 verwendet. Sie arbeitet ebenso nach dem ISOagriNET Standard. Über die in Tabelle 5.17 genannten Konfigurationsparameter hinaus, bietet sie die Möglichkeit, jedem Sensor oder Verbrauchsmesser Koordinaten (x, y, z) zum Zwecke einer zentimetergenauen Verortung zuzuweisen.

Im Folgenden werden die Inhalte und der Aufbau der **Nachrichten** beider Softwareversionen erläutert, wobei die verschiedenen Versionen der Software als DLG Version bzw. Farming Cell Version bezeichnet werden.

Die DLG zertifizierte Version der HME kann lediglich vier unterschiedliche Nachrichtentypen verschicken. Die Anzahl ergibt sich aus der verwendeten **Entität 101000** namens Klima_Luftdaten (vgl. Tabelle 5.1.1). Die Entität kann Werte für Temperatur, Luftfeuchte, CO_2 und NH_3 aufnehmen.

```
DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090219095515HME:00:60:35:07:ee:01
DN101000009010024000090101317000101000062
VN10100012345:1.2.3                              20090219095515000002345
ZN

DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090219095519HME:00:60:35:07:ee:01
DN10100000901002400009010131700010100006200101001041
VN10100012345:1.2.3                              200902190955190000023457891
ZN

DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090219095524HME:00:60:35:07:ee:01
DN10100000901002400009010131700010100006200101006061
VN10100012345:1.2.3
20090219095524000002345067891
ZN

DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO201020090219095528HME:00:60:35:07:ee:01
DN10100000901002400009010131700010100006200101005060
VN10100012345:1.2.3
20090219095528000002345006789
ZN
```

Der oben abgedruckte Nachrichtensatz umfasst die vier Nachrichten, die zu Testzwecken über das Webinterface der HME versendet werden können.

Die Farming Cell Version verwendet die selbst definierten Entitäten 1 (environment_values) und 2 (meter_values), die in den Tabellen 5.6 und 5.7 definiert sind. Ein Beispiel einer ADIS/ADED Nachricht der Entität 2 findet sich in Kapitel 5.6.3. Folgende Nachricht ist ein Beispiel für eine **Entität 1**, wie sie von der Farming Cell Version der Software verschickt wird.

```
DH990054000000000800090000208000901059080009000030800090000406000900006240
VH990054AGRO201000002010AGRO20102009072117590З HME:00:60:35:07:ee:01
DN0000010090100240000901013170000000010420000000204200000003042 00101005060
VN000001                                                           0:3.1.0
2009072117590300003440314009200052 8
ZN
```

Der genutzte Header 990054 (isoagrinet_header) und dessen Aufbau werden im Kapitel 2 detailliert vorgestellt und daher an dieser Stelle nicht erläutert. Die Zeile ZN zeigt das physische Ende der Nachricht an. Die eigentlichen Informationen sind in den zwei mit Definition Normal Data (DN) und Value Normal Data (VN) beginnenden Zeilen enthalten. Die DN Zeile definiert den Aufbau der darauf folgenden VN Zeile.

Der Aufbau der **Definitionszeile** gestaltet sich wie folgt (Leerzeichen zwischen den Nachrichtenabschnitten nachträglich eingefügt):

```
DN000001 00901002400 00901013170 00000001042 00000002042 00000003042
00101005060
```

Tabelle 5.18: Aufbau einer ADIS/ADED Definitionszeile

Nachrichtenabschnitt	Erläuterung
000001	Entität 1 (environment_values)
00901002 40 0	Item 901002 (location_id); Länge 40, Auflösung 0
00901013 17 0	Item 901002 (timestamp); Länge 17, Auflösung 0
00000001 04 2	Item 1 (coordinate_x); Länge 4, Auflösung 2
00000002 04 2	Item 2 (coordinate_y); Länge 4, Auflösung 2
00000003 04 2	Item 3 (coordinate_z); Länge 4, Auflösung 2
00101005 06 0	Item 101005 (CO_2); Länge 6, Auflösung 0

Der Aufbau der dazu gehörenden **Nutzdatenzeile** ist der folgende (Leerzeichen zwischen den Nachrichtenabschnitten nachträglich eingefügt):

```
VN000001 0:3.1.0                                    20090721175903000 0344
0314 0092 000528
```

Tabelle 5.19: Aufbau einer ADIS/ADED Nutzdatenzeile

Nachrichtenabschnitt	Erläuterung
000001	Entität 1 (environment_values)
0:3.1.0	Wert für Item 901002 (location_id); Betrieb 0, Stall 3, Abteil 1, Bucht 0
20090721175903000	Wert für Item 901002 (timestamp); 21.07.2009 17:59:03
0344	Wert für Item 610035 (coordinate_x); 3,44 m
0314	Wert für Item 610035 (coordinate_y); 3,14 m
0092	Wert für Item 610035 (coordinate_z); 0,92 m
000528	Wert für Item 610035 (CO_2); 528 ppm

Es ist nicht erforderlich, alle zu einer Entität gehörenden Items zu verwenden. Als optional gekennzeichnete Items (vgl. Tabelle 5.6) werden nur dann durch die HME verschickt, wenn für diese Messwerte vorliegen. Ebenso ist es möglich, Items deren Messwerte nicht vorliegen, dennoch zu verschicken und den Messwert durch Fragezeichen zu ersetzen (vgl. ISO, 2009 und Kapitel 5.3.2). Letztgenanntes Vorgehen kann jedoch die Netzwerklast und den Verarbeitungsaufwand auf Empfängerseite erhöhen.

Die HME sendet ausschließlich Multicast Nachrichten. Gemäß ISOagriNET Standard ist bei diesem Nachrichtentyp auf den Versand eines **Headers** zu verzichten (ISO, 2009). Da ohne diesen die Zuordnung der übermittelten Daten zu ihrer Quelle nicht möglich ist (Absenderkennung im Header), werden von der HME ausschließlich Nachrichten mit Header versendet. Kapitel 7.2.1 geht auf diesen Aspekt detaillierter ein.

5.6.5 ISOagriNET Parser

Innerhalb des Netzwerkes der Farming Cell werden durch

- den Ethernetbox Service,
- die HME und
- den Möller ISOagriNET-LON-Adapter

laufend Multicast Nachrichten verbreitet.

Der Notwendigkeit, diese Nachrichten zu verarbeiten und die enthaltenen Informationen in der Farming Cell Datenbank abzulegen, trägt der sogenannte ISOagriNET Parser Rechnung. Er nimmt alle an die ISOagriNET **Multicast Gruppe** gesendeten Nachrichten auf und prüft deren Korrektheit unter Verwendung des ADED Data Dictionaries, welches in Form einer dateibasierten SQLite Datenbank vorliegt. Syntaktisch korrekte Nachrichten werden zerlegt und extrahierte Informationen in die **Datenbank** geschrieben. Die folgende Abbildung zeigt beispielhaft den Aufruf des ISOagriNET Parsers und dessen Ausgaben.

```
Terminal
Datei  Bearbeiten  Ansicht  Terminal  Reiter  Hilfe
440-vth142:/home/farmingcell/Desktop/ADIS_Receiver# java -jar ADIS_ADED_Receiver
.jar
22.10.2009 16:07:59 MulticastServer <init>
INFO: New MulticastServer for group 224.111.234.123:2434 is running.
processData: {3=DN000002009010024000090101317000000001042000000002042000000030420
0000006082, 4=VN0000020:3.0.0                                   200910221608010001
0610190018300000000}
line: DN00000200901002400009010131700000000104200000000204200000000304200000006082
line: VN0000020:3.0.0                                   20091022160801000106101900
18300000000
22.10.2009 16:08:05 db.MeasurementAccess insert
INFO: Storing data of entity 2 from sender Ebox:00-90-c2-d2-26-af.
processData: {3=DN000002009010024000090101317000000001042000000002042000000030420
0000006082, 4=VN0000020:3.0.0                                   200910221608020001
0470190018300000000}
line: DN00000200901002400009010131700000000104200000000204200000000304200000006082
line: VN0000020:3.0.0                                   20091022160802000104701900
18300000000
22.10.2009 16:08:05 db.MeasurementAccess insert
INFO: Storing data of entity 2 from sender Ebox:00-90-c2-d2-26-af.
processData: {3=DN000002009010024000090101317000000001042000000002042000000030420
0000007082, 4=VN0000020:3.0.0                                   200910221608020001
3650168001600000000}
line: DN00000200901002400009010131700000000104200000000204200000000304200000007082
line: VN0000020:3.0.0                                   20091022160802000136501680
01600000000
22.10.2009 16:08:06 db.MeasurementAccess insert
INFO: Storing data of entity 2 from sender Ebox:00-90-c2-d2-26-af.
processData: {3=DN00000100901002400009010131700000000104200000000204200000000304200000010031, 4=VN0000010:3.1
.0                              20091022160910000000000000000016}
line: DN00000100901002400009010131700000000104200000000204200000000304200000010031
line: VN0000010:3.1.0                                   20091022160910000000000000000016
22.10.2009 16:09:14 db.MeasurementAccess insert
INFO: Storing data of entity 1 from sender Ebox:00-90-c2-d2-24-64.
} 4=TN1010000:1.2.0                                   2009102216100600  1840????
line: DN101000009010024000090101317000101000062001010010 41
line: VN1010000:1.1.0                                   2009102216100600  1930????
22.10.2009 16:09:14 parser.ADIS_parser parseVNline
INFO: Skipping 101001/humidity with value ????.
22.10.2009 16:09:14 db.MeasurementAccess insert
INFO: Storing data of entity 101000 from sender unknown.
line: VN1010000:1.2.0                                   2009102216100600  1840????
22.10.2009 16:09:14 parser.ADIS_parser parseVNline
INFO: Skipping 101001/humidity with value ????.
22.10.2009 16:09:14 db.MeasurementAccess insert
INFO: Storing data of entity 101000 from sender unknown.
line: TN
processData: {3=DN00000100901002400009010131700000000104200000000204200000000304200101004060, 4=VN0000010:3.1
.0                              200910221614130000294034400092000043}
line: DN00000100901002400009010131700000000104200000000204200000000304200101004060
line: VN0000010:3.1.0                                   200910221614130000294034400092000043
22.10.2009 16:10:33 db.MeasurementAccess insert
INFO: Storing data of entity 1 from sender HME:00:60:35:07:ed:aa.
processData: {3=DN0000010090100240000901013170001010000062, 4=VN0000010:0.0.0
  20091022161203000001281}
line: DN0000010090100240000901013170001010000062
line: VN0000010:0.0.0                                   20091022161203000001281
22.10.2009 16:10:48 db.MeasurementAccess insert
INFO: Storing data of entity 1 from sender HME:00:60:35:07:ee:01.
```

Abbildung 5.16: ISOagriNET Parser

Um dem Benutzer die Überwachung der innerhalb der Multicastgruppe publizierten Nachrichten zu ermöglichen, gibt der ISOagriNET Parser den Nutzdatenanteil einer empfangenen Nachricht und darauf folgend Statusmeldungen zu dessen Verarbeitung aus.

Anhand des folgenden Beispiels werden die **Ausgaben** während der Verarbeitung einer von der HME publizierten Nachricht erläutert.

```
processData:                {3=DN000001009010024000090101317000101001041,
4=VN0000010:0.0.0                                 20091023151216000O855}
line: DN000001009010024000090101317000101001041
line: VN0000010:0.0.0                                 20091023151216000O855
23.10.2009 15:10:59 db.MeasurementAccess insert
INFO: Storing data of entity 1 from sender HME:00:60:35:07:ee:01.
```

Zeilenweise Erläuterung der Statusmeldungen

1. Eine mit processData: eingeleitete Ausgabe enthält den **Nutzdatenteil** der empfangenen Nachricht mit vorangestellter Zeilennummer. Auf die Ausgabe der beiden Header Zeilen wird verzichtet.

```
processData:                {3=DN000001009010024000090101317000101001041,
4=VN0000010:0.0.0                                 20091023151216000O855}
```

2. Eine Ausgabe in der Form line:[Nachrichtenzeile] zeigt an, dass die entsprechende Zeile derzeit abgearbeitet wird. Die Zeile betreffende **Hinweis- und Fehlermeldungen** wie in Abbildung 5.16 zu sehen folgen unmittelbar (s.u. Erläuterung der durch den Möller ISOagriNET-LON-Adapter verursachten Warnmeldungen).

```
line: DN000001009010024000090101317000101001041
line: VN0000010:0.0.0                                 20091023151216000O855
```

3. Es folgt die Meldung der Klasse für **Datenbankzugriffe**, dass Daten geschrieben werden. Hinweis- und Fehlermeldungen folgen auch hier unmittelbar.

```
23.10.2009 15:10:59 db.MeasurementAccess insert
INFO: Storing data of entity 1 from sender HME:00:60:35:07:ee:01.
```

Erläuterung der durch den Möller ISOagriNET LON Adapter verursachten Warnmeldungen

Die von der Lüftungssteuerung kommenden Nachrichten weisen eine Besonderheit auf. Liegt für ein Item kein Messwert vor, wird es dennoch verschickt und Fragezeichen werden als Platzhalter gesetzt. Dieses Vorgehen ist standardkonform, erhöht jedoch die Netzlast und den Verarbeitungsaufwand ohne Mehrwert zu bieten. Der ISOagriNET Parser ist in der Lage, solche Nachrichten zu verarbeiten, gibt jedoch während der Verarbeitung Warnhinweise aus.

Die folgenden Zeilen verdeutlichen den geschilderten Sachverhalt anhand einer Nutzdatenzeile.

1. Die zu verarbeitende Nutzdatenzeile wird ausgegeben.

```
line: VN1010000:1.1.0                                    20091023150231000
1860????
```

2. Das Item 101001 (humidity) enthält als Wert ???? und wird daher übersprungen.

```
23.10.2009 15:01:37 parser.ADIS_parser parseVNline
INFO: Skipping 101001/humidity with value ????.
```

3. Die Daten werden regulär in die Datenbank geschrieben, wobei Item 101001 (humidity) NULL gesetzt wird.

```
23.10.2009 15:01:37 db.MeasurementAccess insert
INFO: Storing data of entity 101000 from sender unknown.
```

Da der ISOagriNET Parser eingehende Nachrichten in Echtzeit analysieren und speichern muss, ist der Zahl verarbeitbarer Nachrichten pro Zeiteinheit eine Grenze gesetzt. Zwar wurde diese bisher nicht erreicht, ein **Optimierungsvorschlag** wird dennoch in Kapitel 7.2.3 vorgestellt.

5.6.6 Datenbank

Professionelle Datenbankanwender setzten bisher zumeist auf etablierte proprietäre Produkte wie Oracle Database oder DB2 der Firma IBM. Der Trend, Open-Source Software zu verwenden, um Kosten zu sparen und auf die Unterstützung der großen Internetgemeinde zurückgreifen zu können, hat die Verbreitung von Open-Source-Datenbanken begünstigt. Ein qualitativer Vergleich der verschiedenen verfügbaren Datenbanken, Open wie auch Closed-Source, ist kaum möglich. Jedes Datenbanksystem hat individuelle Stärken (vgl. HORSTMANN, 2006).
Die kostenmotivierte Entscheidung, eine Open-Source Variante für die Farming Cell zu wählen, sowie die Entscheidung, eine relationale Datenbank zu verwenden, schränkte die Auswahl ein. Zur Gruppe möglicher Datenbanken zählten MySQL und PostgreSQL, die aufgrund vorangegangener Erfahrungen aus anderen Projekten für das Projekt Farming Cell in Betracht kamen.

Die Wahl fiel auf **MySQL**. Bei MySQL handelt es sich um ein als Open-Source-Software sowie als kommerzielle Enterpriseversion verfügbare relationale Datenbank

entwickelt von der Firma Sun Microsystems. Die ausschlaggebenden Gründe für die Wahl waren die folgenden:

- XAMPP (vgl. Kapitel 4.2) ermöglicht das komfortable Aufsetzen eines Datenbanksystems mit MySQL als Datenbank auf verschiedenen Betriebssystemen und dessen komfortable Administration mit dem Datenbankmanagementsystem phpMyAdmin.
- MySQL bietet das Werkzeug der Views (temporär erzeugte Tabellen), welche tabellenübergreifende Zusammenhänge abbilden.
- Das Rechenzentrum der Universität Hohenheim und Firmen wie die Amazon Tochter Amazon-Web-Services[23] (AWS) vermieten MySQL Datenbanken. Ein Backup auf diese ist damit möglich.
- Die Replikationsmöglichkeiten von MySQL erlauben das parallele Befüllen einer zweiten Datenbank zu Sicherungszwecken (vgl. KERSKEN, 2009).

Das Rechenzentrum der Universität Hohenheim bietet derzeit ausschließlich MySQL Datenbanken (vgl. ANCUTICI, 2009) an. Im Hinblick auf das in Kapitel 7.2.3 angeregte Redesign des Datenmodells, könnte die Wahl eines anderen Datenbanksystems eine Alternative darstellen.

Datenbank Design

Die Datenbank als zentrales Element der Farming Cell hält alle den Schweinmastprozess betreffenden Informationen vor. Die Gesamtheit der Daten kann den folgenden drei **Cluster** zugeordnet werden:

1. Sensoren, Verbrauchsmesser und Anlagen betreffende Daten
2. Tierspezifische Daten
3. Andere Daten

Die einzelnen Cluster und deren Berührungspunkte werden im Folgenden vorgestellt.

Cluster 1

Der Großteil der in diesem Cluster befindlichen Daten stammt von **Sensoren** und **Verbrauchsmessern**. Ihnen ist gemein, dass sie Messwerte besitzen, welchen

[23] Internetpräsenz von Amazon-Web-Services LLC (AWS): http://aws.amazon.com

wiederum Zeitpunkte zugeordnet sind. Darüber hinaus ist eine Verortung der Sensoren und Verbrauchsmesser möglich, wobei diese in verschiedenen Detailgraden erfolgen kann (siehe unten).
Ein weiteres Datentaufkommen dieses Clusters wird durch die **Lüftungssteuerung** verursacht. Diese liefert neben Messwerten regelmäßig die Entität 101005 (Klima_Tierdaten). Da die Lüftungssteuerung keine Kenntnisse über Tieranzahl und die anderen Parameter dieser Entität hat, ist derzeit kein Informationsgehalt gegeben. Dennoch ist die Datenbank für die Aufnahme der Entität 101005 vorbereitet (Tabelle entity_101005).

Weitere diesem Cluster zugeordnete Tabellen sind:

- entity_1 (environment_values)
- entity_2 (meter_values)
- entity_101000 (Klima_Luftdaten)
- entity_101001 (Klima_Außen)

Die Tabellen entity_1, entity_2, entity_101000 und entity_101001 ermöglichen die Aufnahme aller im Gesamtsystem anfallenden Werte von Sensoren und Verbrauchsmessern. Der Aufbau dieser drei **Tabellen** ist nahezu identisch mit dem des ADED Datadictionaries. Sie alle besitzen die Felder farm, stable, compartment, pen, timestamp und sender. Die folgende Tabelle erläutert die Felder.

Tabelle 5.20: Standard-Felder der Entity Tabellen

Feld	Erläuterung
farm	Betriebsnummer (lokaler Betrieb = 0)
stable	Stallnummer
compartment	Abteilnummer
pen	Buchtnummer
timestamp	Messzeitpunkt
sender	Eindeutige Kennung des Gerätes, welches die Entität geschickt hat

Der übermittelte Messwert wird in einem weiteren Feld abgelegt, dessen Benennung seinem Typ entspricht. Die zulässigen Typen sind Tabelle 5.17 zu entnehmen.
Die Tabellen entity_1 und entity_2, welche für die Aufnahme der beiden Farming Cell internen Entitäten vorhanden sind, besitzen darüber hinaus die Felder coordinate_x, coordinate_y und coordinate_z.

Tabelle 5.21: Koordinatenfelder der Entity Tabellen

Feld	Erläuterung
coordinate_x	X Koordinate des Messpunktes
coordinate_y	Y Koordinate des Messpunktes
coordinate_z	Z Koordinate des Messpunktes

Die Notwendigkeit der **Koordinatenfelder** liegt in der Anforderung begründet, Messpunkte definieren zu können (siehe Erläuterungen zur Tabelle measuring_points unten), denen die in den Tabellen entity_1 und entity_2 enthaltenen Einträge zugeordnet werden können.
Die in den Tabellen des Clusters abgelegten Messdaten haben ihren Ursprung in den drei Quellen ISOagriNET-LON-Adapter, Ethernetbox Service und HME Software. Wenn möglich werden die Entitäten 1 und 2 für die Datenübertragung verwendet, um Messkoordinaten definieren zu können. Die folgende Tabelle stellt dar, welche Entitäten jede Quelle publizieren kann.

Tabelle 5.22: Publizierten Entitäten nach Ursprung

	ISOagriNET-LON-Adapter	Ethernetbox Service	HME Software
entity_1		x	x (1)
entity_2		x	x (1)
entity_101000	x		x (2)
entity_101001	x		
entity_101005	x		

(1) Farming Cell Version der HME Software

(2) DLG Version der HME Software

Die der Lüftungssteuerung ausgenommen, sind alle gelieferten Messwerte zu zentimetergenau definierbaren Messstellen zuzuordnen. Damit diese **Verortung** datenbankseitig erfolgen kann, existieren die Tabellen location und measuring_points. Die Pflichtfelder farm, stable, compartment und pen (ISO, 2009) jeder Nutzdatenzeile einer ADIS/ADED Nachricht ermöglichen deren Zuordnung zu einem in der Tabelle location hinterlegen Ort. Im Rahmen des Projektes zulässige Orte nennt die folgende Tabelle 5.23.

Tabelle 5.23: Inhalt der Tabelle location[1]

farm_number	stable_number	compartment _number	pen_number	description
0	3	1	1	Bucht 1
0	3	1	2	Bucht 2
0	3	2	3	Bucht 3
0	3	2	4	Bucht 4
0	3	1	0	Abteil 1
0	3	2	0	Abteil 2
0	0	0	0	Außen/Vorraum

[1] Stand 27.11.2009

Sind in einer Nachricht darüber hinaus die optionalen Koordinatenfelder coordinate_x, coordinate_y und coordinate_z gesetzt, kann eine Zuordnung des Messwertes zu einer in der Tabelle measuring_points hinterlegten Messstelle erfolgen. Die Tabelle measuring_points bildet die aus Koordinaten und einem Gültigkeitszeitraum modellierten Messstellen ab. Das den Koordinaten zugrunde liegende Koordinatensystem des Versuchstalls auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof zeigt die Abbildung 5.8.

Sind die Lokationsfelder einer Nachricht falsch oder unvollständig belegt, ist die Nachricht ungültig und würde vom ISOagriNET Parser (vgl. Kapitel 5.6.5) mit einer entsprechenden Fehlermeldung verworfen. Mögliche Kombinationen nennt Tabelle 5.24.

Tabelle 5.24: Lokations- und Koordinatenfelder

Lokationsfelder	Koordinatenfelder	Erläuterung
enthalten Werte	enthalten Werte	Zuordnung zu einer zentimetergenauen Messstelle möglich, sofern diese in Tabelle measuring_points eingetragen ist. Ist keine Messstelle hinterlegt, bestimmen die Lokationsfelder die Genauigkeit
enthalten Werte	enthalten keine Werte (alle Felder NULL oder alle Felder 0)	Lokationsfelder bestimmen Genauigkeit (vgl. zulässig Lokationen in Tabelle 5.23).
enthalten keine Werte	enthalten keine Werte	Nachricht ungültig

Eine weitere diesem Cluster angehörende Tabelle ist die **Tabelle sensors**. Sie dient dem Zweck, die im Gesamtsystem zulässigen Sensor- und Verbrauchsmessertypen sowie an sie gebundene Informationen zu hinterlegen. Die in Kapitel 5.6 vorgestellten Programme Info Mailer und Reporting Applikation nutzen die in dieser Tabelle hinterlegten Informationen. Sie enthält Angaben zu Ober- und Untergrenzen von Messsollbereichen, die Einheit der Messwerte und die deutsche Bezeichnung des Sensors, wie sie dem Benutzer anzuzeigen ist.

Daten der Fütterung werden ebenfalls in Cluster 1 abgelegt. Die **Tabelle feed_consumption** bietet die hierfür erforderliche Struktur. Gespeichert werden die täglichen Ventilverbräuche aller Buchten und die Tieranzahl (vgl. Kapitel 5.6.1 Schauer Service).

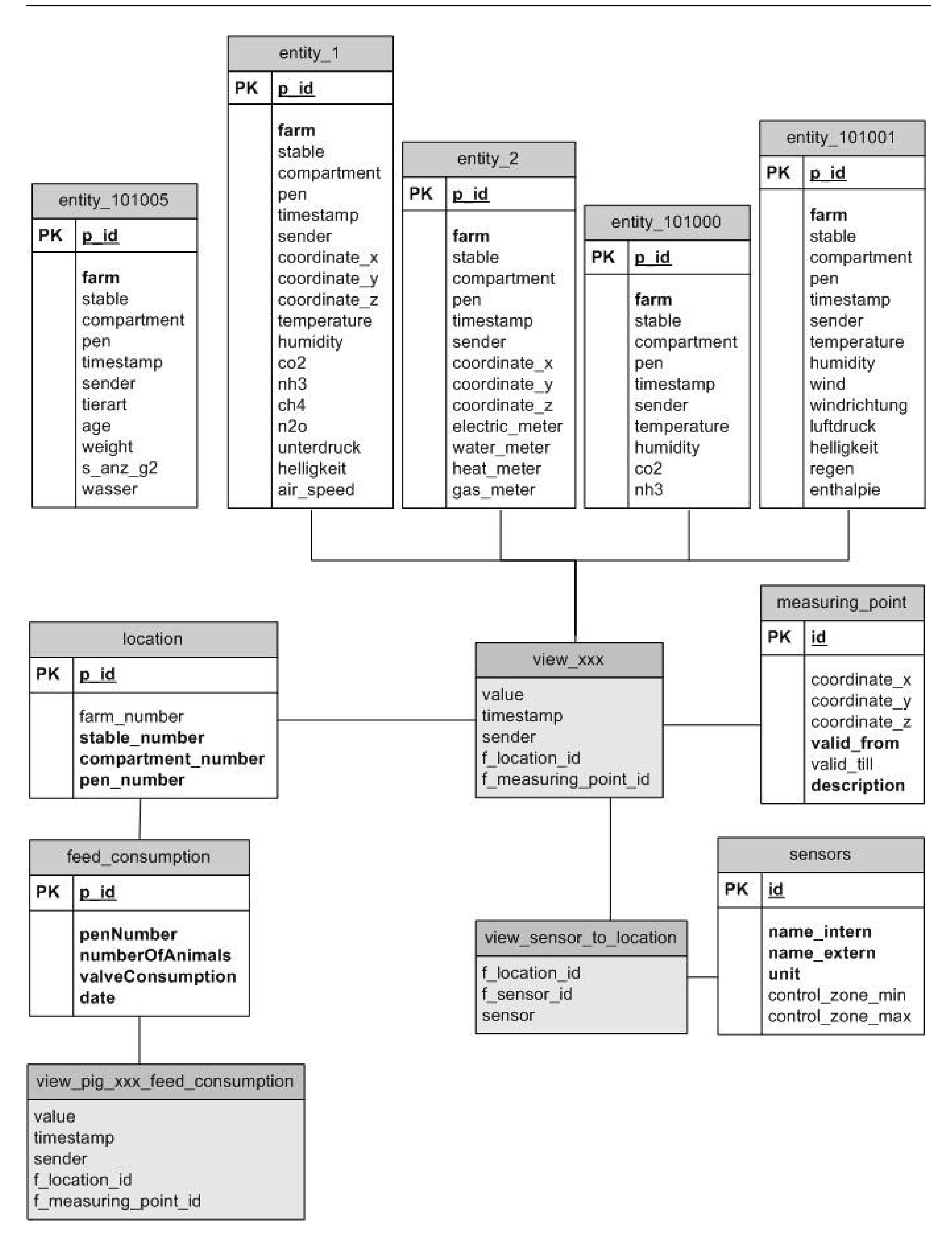


Abbildung 5.17: Datenbank Schema - Cluster 1 - Mess- und Anlagendaten

Zum Zwecke besserer Auswertungsmöglichkeiten wurden in der Datenbank der Farming Cell Sichten (engl. Views) angelegt. **Views** sind wie Tabellen zu nutzen, bilden jedoch eine Teilmenge bestimmter Daten der Datenbank nur temporär ab. Auf

diese Weise ist es möglich, ausgewählte Teile verschiedener Tabellen miteinander in Beziehung zu setzen und darzustellen.
In der Datenbank existieren zwei Arten von Views. Zum einen diejenigen, die Rohdaten darstellen, zum anderen solche, die Stundensummen oder Stundendurchschnittswerte errechnen und darstellen.

Die Benennung der Views folgt dem Schema view_[Messwerttyp] für diejenigen Views, welche Rohdaten enthalten. Diejenigen die Stundensummen oder Stundenmittel bereitstellen lauten auf view_avg_[Messwerttyp].

Beispiel:
view_nh3 - enthält Rohdaten
view_avg_nh3 - enthält Stundenmittel für jeden Messort bzw. jede Messstelle (sofern Rohdaten vorhanden)

Der Aufbau aller Views ist identisch. Folgende Felder sind in jeder View vorhanden.

Tabelle 5.25: Felder der Datenbank Views

Feld	Erläuterung
value	Der Messwert.
timestamp	Der Zeitpunkt an dem der Messwert erfasst wurde (Rohdaten) oder die Stunde für die eine Mittelwert / die Summe berechnet wurde.
sender	Der Name oder Identifier des Gerätes oder der Software das/die der Ursprung eines Wertes ist.
f_location_id	Die ID des den Messort repräsentierenden Eintrags aus der Tabelle location.
f_measuring_point_id	Die ID des den Messpunkt repräsentierenden Eintrags aus der Tabelle measuring_point.

Bei den bisher nicht erwähnten Feldern f_location_id und f_measuring_point_id handelt es sich um Fremdschlüssel (Referenzen auf Primärschlüssel anderer Tabellen). Ein Eintrag in f_location_id referenziert auf einen Eintrag in der Tabelle location, wodurch die Zuordnung des Messwertes zu einer Lokation wie einem Abteil oder einer Bucht erfolgt. Die zweite Referenz ist f_measuring_point_id und verweist auf eine Messstelle in der Tabelle measuring_points.

Da der Möller ISOagriNET LON Adapter der Lüftungssteuerung bis zum Ende des Implementierungsprozesses nur in der Lage war, feste Werte für zwei der Felder (farmnumber = Testkunde, stable = 1) zu senden, werden die Attribute stable_number und farm_number für die View Erzeugung nicht berücksichtigt. Für die Farming Cell ist dies solange unproblematisch, wie sie nicht auf weitere Ställe oder auf andere Betriebe ausgedehnt wird.
Mit einer neuen Firmware Version hat die Firma Möller diese Beschränkung im Oktober 2009 beseitigt. Die im Falle einer Erweiterung der Farming Cell notwendige Anpassung der Views ist nicht mehr erfolgt.

Folgende dem vorgestellten Schema entsprechenden Views existieren:

Tabelle 5.26: Views in Cluster 1

Views mit Rohdaten	Views mit aggregierten Daten
view_temperature	view avg_temperature
view_humidity	view avg_humidity
view_co2	view avg_co2
view_nh3	view avg_nh3
view_ch4	view avg_ch4
view_n2o	view avg_n2o
view_helligkeit	view avg_helligkeit
view_unterdruck	view avg_unterdruck
view_air_speed	view avg_air_speed
view_wind	view avg_wind
view_windrichtung	view avg_windrichtung

view_luftdruck	view avg_luftdruck
view_regen	view avg_regen
view_enthalpie	view avg_enthalpie
view_electric_meter	view avg_electric_meter
view_water_meter	view avg_water_meter
view_heat_meter	view avg_heat_meter
view_gas_meter	view avg_gas_meter

Des Weiteren existieren in Cluster 1 folgende, über die Darstellung von Messwerten hinausgehende Views:

Tabelle 5.27: Weitere Views in Cluster 1

Name der View	Erläuterung
view_sensor_to_location	Enthält alle Kombinationen von ID aus Tabelle sensors und ID aus Tabelle Location zu denen Werte erfasst werden.
view_pig_total_group_feed_consumption	Enthält den tageweisen Futterverbrauch einer gesamten Gruppe / Bucht
view_pig_avg_pig_feed_consumption	Enthält den durchschnittlichen tageweisen Einzeltierfutterverbrauch einer Gruppe / Bucht

Die für das Anlegen aller Views verwendeten SQL Ausdrücke finden sich im Anhang A dieser Arbeit.

Cluster 2

Im zweiten Cluster sind die Masttiere modelliert und die mit Ihnen verbundenen Informationen hinterlegt. Dies schließt die Zuordnung der Tiere zu Aufenthaltsorten sowie Daten zu Wiegungen und Medikamentierungen ein. Die Modellierung erfolgte mit dem Fokus auf Mastschweine, ist jedoch generisch (vgl. GROENEVELD, 2002)

und daher mit wenigen Änderungen auch für andere Lebensabschnitte und Tierarten nutzbar.
Das Mastschwein und seine Eigenschaften werden mithilfe dreier Tabellen modelliert:

- animal
- boar
- race

Die **Tabelle animal** besitzt unter anderem die in der folgenden Tabelle erläuterten Felder und ist in der Lage, Daten tierindividuell zu halten (p_electronical_id ist Primärschlüssel).

Tabelle 5.28: Attribute der Tabelle animal (Auszug)

Feld	Erläuterung
p_electronical_id	Nummer der elektronischen Ohrmarke
mother_no	Nummer der Muttersau
piglet_no	Nummer die das Tier als Ferkel hatte
date_of_birth	Geburtsdatum
sex	Geschlecht

Die Nummer der elektronischen Ohrmarke ausgenommen, stammen die Daten aus der Managementsoftware Supersau der Firma CLAAS Agrosystems GmbH & Co. KG (ehemals AGROCOM GmbH & Co. Agrarsystem KG), die im vorgelagerten Ferkelproduktionsprozess im Einsatz ist. Der dort gepflegte Tierbestand wird mittels Excel Datei Import in die Tabelle temp_anmial der Farming Cell Datenbank überführt (vgl. Kapitel 5.6.8). Während des Einstallens stehen auf diese Weise die Datensätze aller potentiellen Masttiere zur Verfügung.
Zusätzlich zu den genannten Daten sind in der Software Supersau Informationen über Abstammung und Rasse hinterlegt. Diese werden ebenfalls übertragen und in den beiden **Tabellen boar** (engl. Eber) und **race** (engl. Rasse) in der Datenbank der Farming Cell abgelegt. In den Tabellen temp_animal und animal sind Fremdschlüssel vorhanden, die auf Inhalte dieser beiden Tabellen referenzieren.

Tabelle 5.29: Fremdschlüssel der Tabellen temp_animal und animal

Feld	Erläuterung
f_father_id	Referenz auf einen Eintrag in der Tabelle boar.
f_race_id	Referenz auf einen Eintrag in der Tabelle race.

Ein einzustallendes Tier kann anhand seiner Tätowiernummer, die sich aus der Mutternummer und der Ferkelnummer zusammensetzt, identifiziert werden. Dieser wird beim Einstallprozess die Nummer der neu eingezogenen RFID Ohrmarke zugeordnet. Anschließend wird diese zusammen mit den Einzeltierdaten aus Tabelle temp_animal in die Tabelle animal übertragen.
Eine detaillierte Erläuterung der Prozesse Dateiimport und Einstallen unter Verwendung der entwickelten Webapplikation findet sich in Kapitel 5.6.8. Für vollständige Strukturdarstellungen der genannten Tabellen siehe Anhang A.

Bereits während des Einstallprozesses erfolgt die Zuordnung des Einzeltiers zu einer Bucht (Tabelle location). Dies geschieht mithilfe der Tabelle animal_to_location, in der einem Tier ein oder mehrere Aufenthaltsorte zugewiesen werden (Fremdschlüsselreferenz auf die Tabelle location). Mehrere Einträge sind möglich, da auf diese Weise Umstallungen darstellbar sind. Die Unterscheidbarkeit der Datensätze und die Nachvollziehbarkeit der Tierbewegungen sind mithilfe der beiden ebenfalls hinterlegten Zeitpunkte für den Beginn und das Ende des Aufenthaltes an einem Ort möglich.

Die Tabelle location, die auch in Cluster 1 für die Ortsbestimmung von Messwerten zum Einsatz kommt, bildet das Bindeglied zwischen den Clustern 1 und 2. Die einem Ort zu einem Zeitpunkt zugeordneten Tiere des einen Clusters sowie die ebenfalls verorteten und mit einem Zeitstempel versehenen Messwerte des anderen sind einander zuordenbar und beispielsweise für Auswertungszwecke nutzbar (vgl. Kapitel 5.6.7 und 6).

Eine weitere in diesem Cluster befindliche Tabelle ist entity_610011, welche in der Lage ist, **Einzeltiergewichte** (RFID Ohrmarkennummer und Gewicht) und ein zugeordnetes Wiegedatum aufzunehmen. Die von der Waagensteuerung

übertragenen Informationen (vgl. Kapitel 5.3.3) stehen nach der Speicherung in der Tabelle entity_610011 automatisch in folgenden Views zur Verfügung.

Tabelle 5.30: Views in Cluster 2

Name der View	Erläuterung
view_pig_total_group_weight	Enthält das Gruppengesamtgewicht und die Tieranzahl je Tiergruppe zu jedem Wiegezeitpunkt
view_pig_avg_pig_weight	Enthält das durchschnittliche Einzeltiergewicht je Tiergruppe zu jedem Wiegezeitpunkt
view_pig_daily_gain	Enthält die durchschnittliche Einzeltiertageszunahme je Tiergruppe zwischen zwei Wiegezeitpunkten
view_last_pig_weight_pen1, view_last_pig_weight_pen2, view_last_pig_weight_pen3, view_last_pig_weight_pen4	Enthalten alle tierindividuellen Gewichte der letzten Wiegung.

Neben Gewichten können dem Einzeltier ebenso Daten zu medizinischen Diagnosen und vorgenommenen Behandlungen zugeordnet werden. Folgende Tabellen ermöglichen dies:

- medical_treatment
- diagnosis
- medicine

Die **Tabelle medical_treatment** bildet das zentrale Element dieser Modellierung. Für jede in das System eingepflegte Behandlung oder Diagnose existiert dort ein Eintrag, der anhand der elektronischen Ohrmarkennummer (f_electronical_animal_id) einem Einzeltier (Tabelle animal) zugeordnet werden kann. Über die beiden weiteren Fremdschlüssel f_diagnosis_id und f_medicine_id erfolgt die Verknüpfung mit Einträgen der **Tabellen medicine und diagnosis**. Der Eintrag für f_medicine_id in

der Tabelle medical_treatment ist optional, da eine Diagnose nicht in jedem Fall eine Medikamentierung nach sich zieht. Die folgende Abbildung zeigt das vollständige Datenbankschema von Cluster 2.

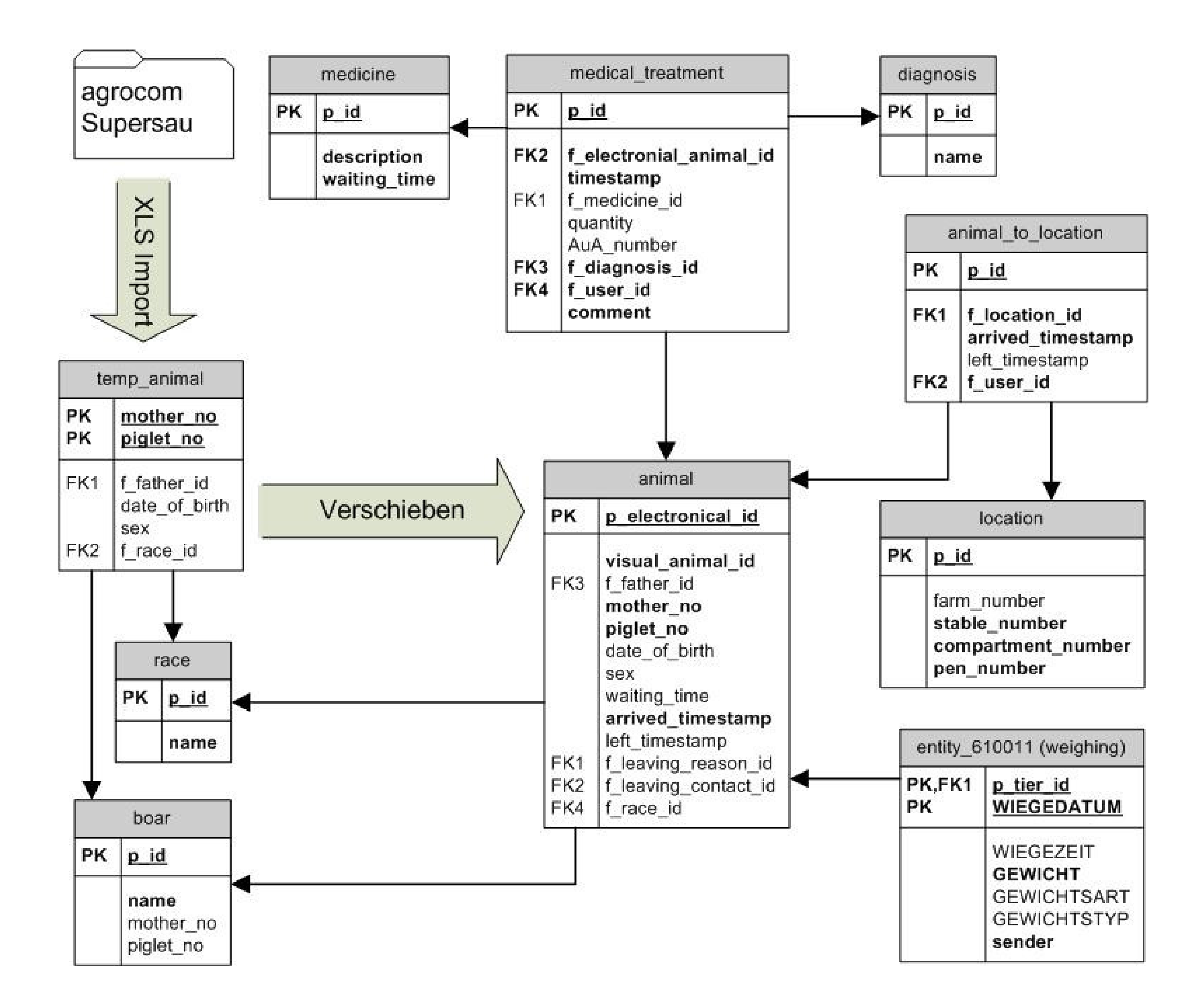


Abbildung 5.18: Datenbank Schema - Cluster 2 - Tierdaten

Die nicht erläuterten Fremdschlüssel f_leaving_reason und f_contact_id der Tabelle animal sowie der Fremdschlüssel f_user_id der Tabellen medical_treatment und animal_to_location referenzieren auf Tabellen, die dem nachfolgend vorgestellten Cluster 3 angehören.

Cluster 3

Dieses Cluster ergänzt das Datenbankschema um Tabellen, die keinem der beiden anderen Cluster unmittelbar zuzuordnen sind (vgl. hervorgehobene Tabellen in Abbildung 5.19). Vielmehr modellieren die Tabellen

- contact,
- leaving_reason,
- user und
- google_chart_url

Objekte, die in verschiedenen Zusammenhängen nutzbar oder erforderlich sind. Wie in der folgenden Abbildung 5.22 zu sehen, werden die **Tabellen contact und leaving reason** über Fremdschlüssel mit der Tabelle animal verknüpft. In der **Tabelle contact** können Kontaktdaten von Wirtschaftspartnern hinterlegt werden, die Tabelle leaving_reason dient der Aufnahme von Gründen für den Abgang von Tieren.

Da zum einen eine Zugangsbeschränkung für die Webapplikation (vgl. Kapitel 5.6.8) mithilfe von Benutzerkonten erforderlich und zum anderen bei ausgewählten Datenbankeinträgen die Zuordnung eines Benutzers sinnvoll ist, existiert die **Tabelle user**.

Die isoliert stehende **Tabelle google_chart_url** wird vom Info Mailer (vgl. Kapitel 5.6.7) als Ablageort für generierte Grafik URI genutzt, damit diese für eine eventuelle spätere Verwendung zur Verfügung stehen.

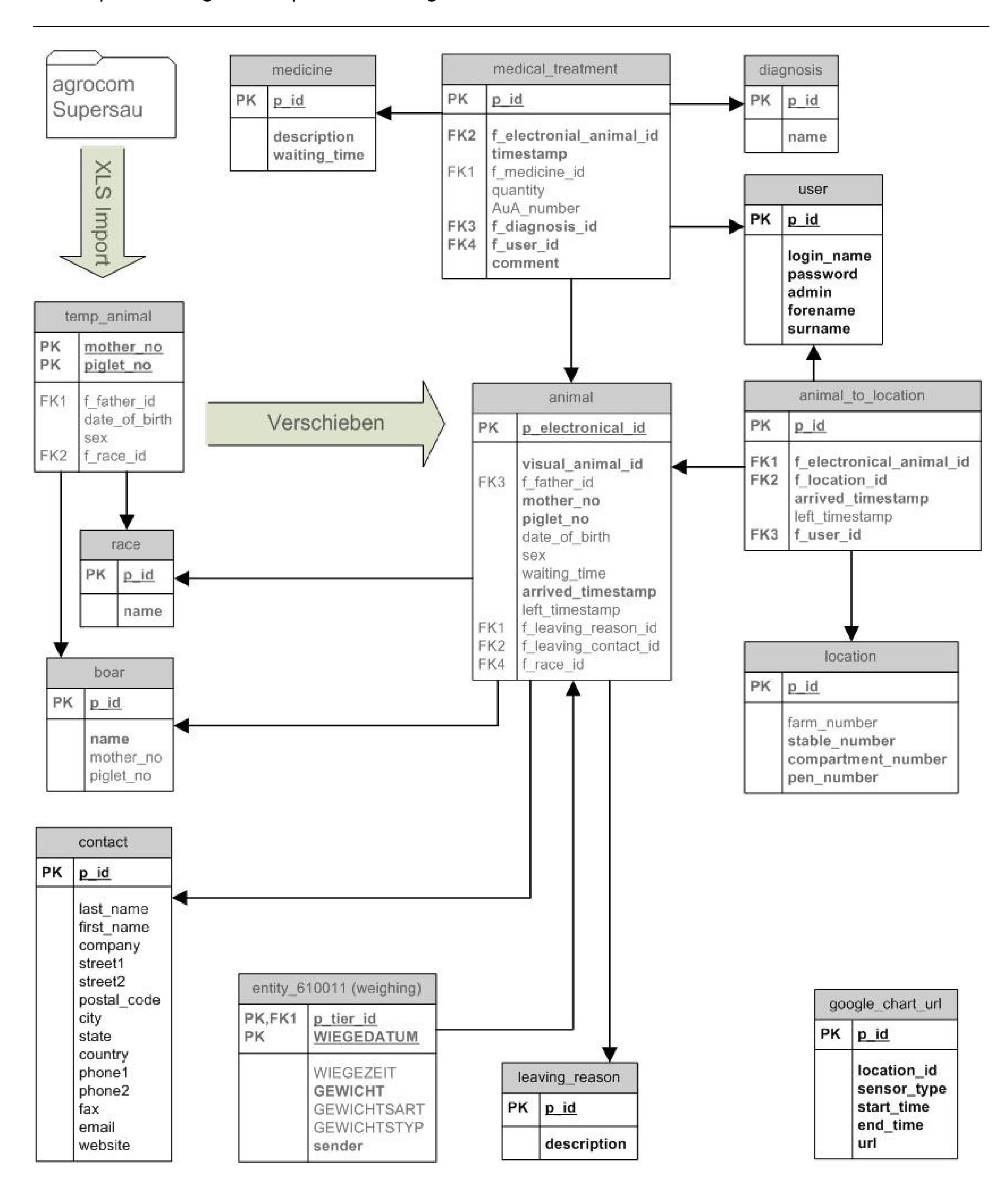


Abbildung 5.19: Datenbank Schema - Cluster 3 - Sonstige

Die folgende Abbildung zeigt das Schema der Farming Cell Datenbank in seiner Gesamtheit. Die Mehrzahl der Attribut- und Tabellennamen sind in englischer Sprache. Die wenigen Ausnahmen in deutscher Sprache sind durch das ADED Data Dictionary vorgegeben.

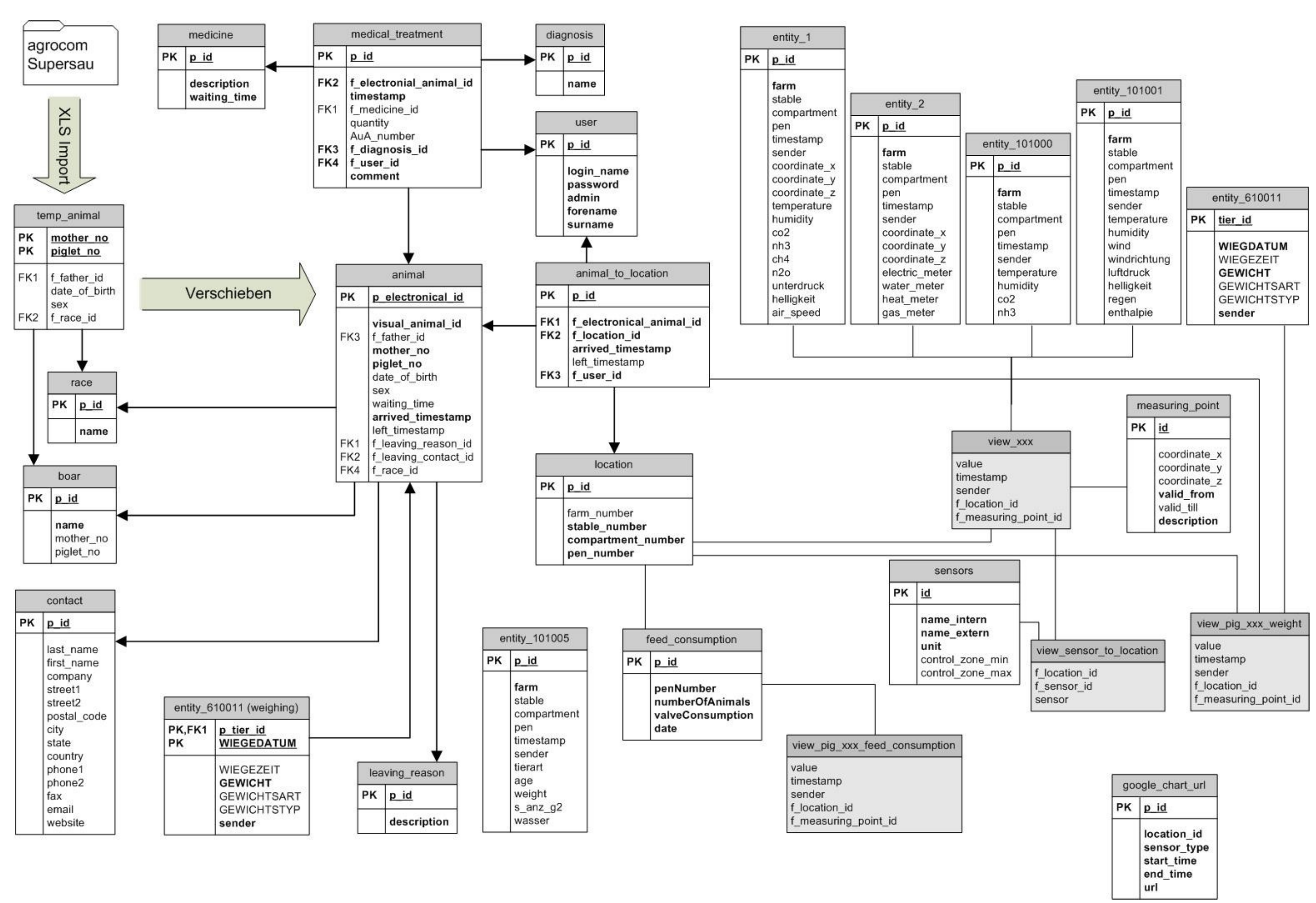
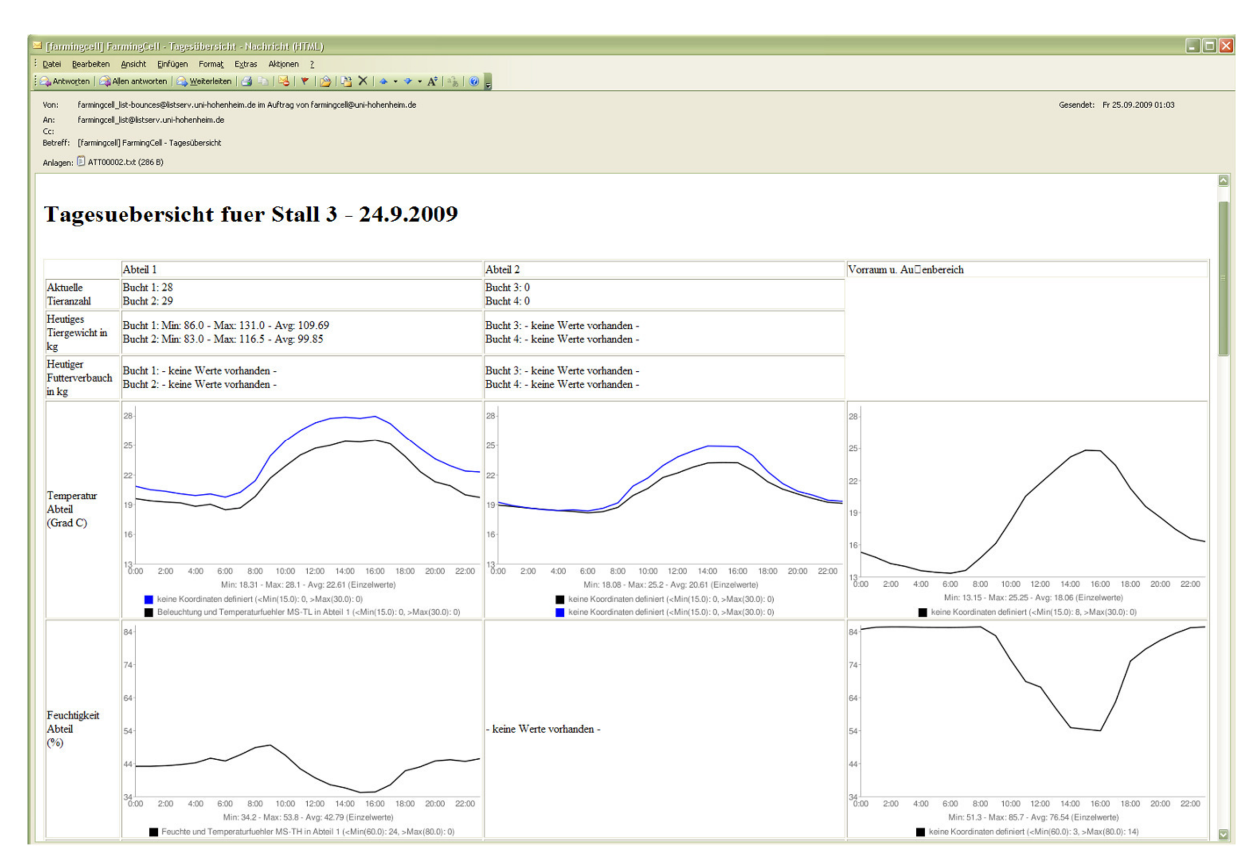


Abbildung 5.20: Schema der Farming Cell Datenbank

5.6.7 Mailingliste und Info Mailer

Eine große Datenmenge wird im Tagesverlauf in der Datenbank der Farming Cell gesammelt. Einerseits ist es möglich, diese Daten aktiv einzusehen (vgl. Kapitel 5.6.9), andererseits werden die Mitglieder einer Mailingliste einmal täglich mit einer E-Mail versorgt, die die wichtigsten **Vortagesinformationen** grafisch aufbereitet darstellt. Die Mailingliste der Farming Cell wird durch das Rechenzentrum der Universität Hohenheim bereitgestellt und trägt den Namen **farmingcell_list**[24].

Das Erzeugen und Versenden der täglichen Status E-Mail wird von der Software Info Mailer übernommen. Sie ist permanent in Betrieb, um einmal am Tag automatisch eine E-Mail aus den in der Farming Cell Datenbank vorhandenen Vortageswerten zu erstellen. Abbildung 5.21 zeigt einen Ausschnitt der E-Mail vom 24.09.2009.


[farmingcell] FarmingCell - Tagesübersicht - Nachricht (HTML)

Von: farmingcell_list-bounces@listserv.uni-hohenheim.de im Auftrag von farmingcell@uni-hohenheim.de
Gesendet: Fr 25.09.2009 01:03
An: farmingcell_list@listserv.uni-hohenheim.de
Cc:
Betreff: [farmingcell] FarmingCell - Tagesübersicht
Anlagen: ATT00002.txt (286 B)

Tagesuebersicht fuer Stall 3 - 24.9.2009

	Abteil 1	Abteil 2	Vorraum u. Au□enbereich
Aktuelle Tieranzahl	Bucht 1: 28 Bucht 2: 29	Bucht 3: 0 Bucht 4: 0	
Heutiges Tiergewicht in kg	Bucht 1: Min: 86.0 - Max: 131.0 - Avg: 109.69 Bucht 2: Min: 83.0 - Max: 116.5 - Avg: 99.85	Bucht 3: - keine Werte vorhanden - Bucht 4: - keine Werte vorhanden -	
Heutiger Futterverbauch in kg	Bucht 1: - keine Werte vorhanden - Bucht 2: - keine Werte vorhanden -	Bucht 3: - keine Werte vorhanden - Bucht 4: - keine Werte vorhanden -	
Temperatur Abteil (Grad C)	Min: 18.31 - Max: 28.1 - Avg: 22.61 (Einzelwerte) keine Koordinaten definiert (<Min(15.0): 0, >Max(30.0): 0) Beleuchtung und Temperaturfuehler MS-TL in Abteil 1 (<Min(15.0): 0, >Max(30.0): 0)	Min: 18.08 - Max: 25.2 - Avg: 20.61 (Einzelwerte) keine Koordinaten definiert (<Min(15.0): 0, >Max(30.0): 0) keine Koordinaten definiert (<Min(15.0): 0, >Max(30.0): 0)	Min: 13.15 - Max: 25.25 - Avg: 18.06 (Einzelwerte) keine Koordinaten definiert (<Min(15.0): 8, >Max(30.0): 0)
Feuchtigkeit Abteil (%)	Min: 34.2 - Max: 53.8 - Avg: 42.79 (Einzelwerte) Feuchte und Temperaturfuehler MS-TH in Abteil 1 (<Min(60.0): 24, >Max(80.0): 0)	- keine Werte vorhanden -	Min: 51.3 - Max: 85.7 - Avg: 76.54 (Einzelwerte) keine Koordinaten definiert (<Min(60.0): 3, >Max(80.0): 14)

Abbildung 5.21: Tägliche Status E-Mail

[24] Die Administration ist unter https://listserv.uni-hohenheim.de/mailman/admin/farmingcell_list möglich. Benutzer der Liste haben Zugriff unter https://listserv.uni-hohenheim.de/mailman/listinfo/farmingcell_list. E-Mails an die Liste sind an farmingcell_list@listserv.uni-hohenheim.de zu adressieren.

Abhängig vom Mastfortschritt und der damit verbundenen Größe der Datenbank, nimmt die E-Mail Generierung viel Zeit in Anspruch (> 1 h nach ca. 100 Masttagen und einer Datenbankgröße > 4 GB). Aus diesem Grunde sollte dieser Vorgang vorzugsweise nachts erfolgen (vgl. Kapitel 5.7). Neben dem Ausführungszeitpunkt existieren weitere **Steuerungsparameter**. Diese sind in der Datei setup.properties hinterlegt.

Inhalt der Datei setup.properties

```
run_Hour = 00
run_Minute = 17
sensor_types_compartment = temperature;humidity;co2;nh3
sensor_types_pen =
meter_types_compartment = water_meter;heat_meter
meter_types_pen = water_meter
mail_recipients = farmingcell_list@listserv.uni-hohenheim.de
```

Die Parameter run_hour und run_Minute bestimmen den Ausführungszeitpunkt. In diesem Beispiel wird täglich um 00:17 mit der Generierung der E-Mail begonnen. Die Parameter sensor_types_compartment, sensor_types_pen, meter_types_compartment und meter_types_pen legen fest, für welche Sensor- und Verbrauchsmessertypen Diagramme erzeugt und in der E-Mail verschickt werden. Zulässig sind alle Werte, für welche in der Datenbank der Farming Cell Views hinterlegt sind.

Beispielsweise definiert die Zeile meter_types_pen = water_meter, dass die Darstellung der Wasserverbrauchswerte auf Buchtenebene in der E-Mail enthalten sein soll. Als Datenbasis dient in einem solchen Fall die View view_avg_water_meter, welche die Stundensummen der einzelnen Wasservolumenmesser enthält. Abbildung 5.22 zeigt, wie eine Darstellung für Verbrauchsmesserwerte, im Vergleich zu der Darstellung von Sensorwerten in Abbildung 5.21, aussieht.

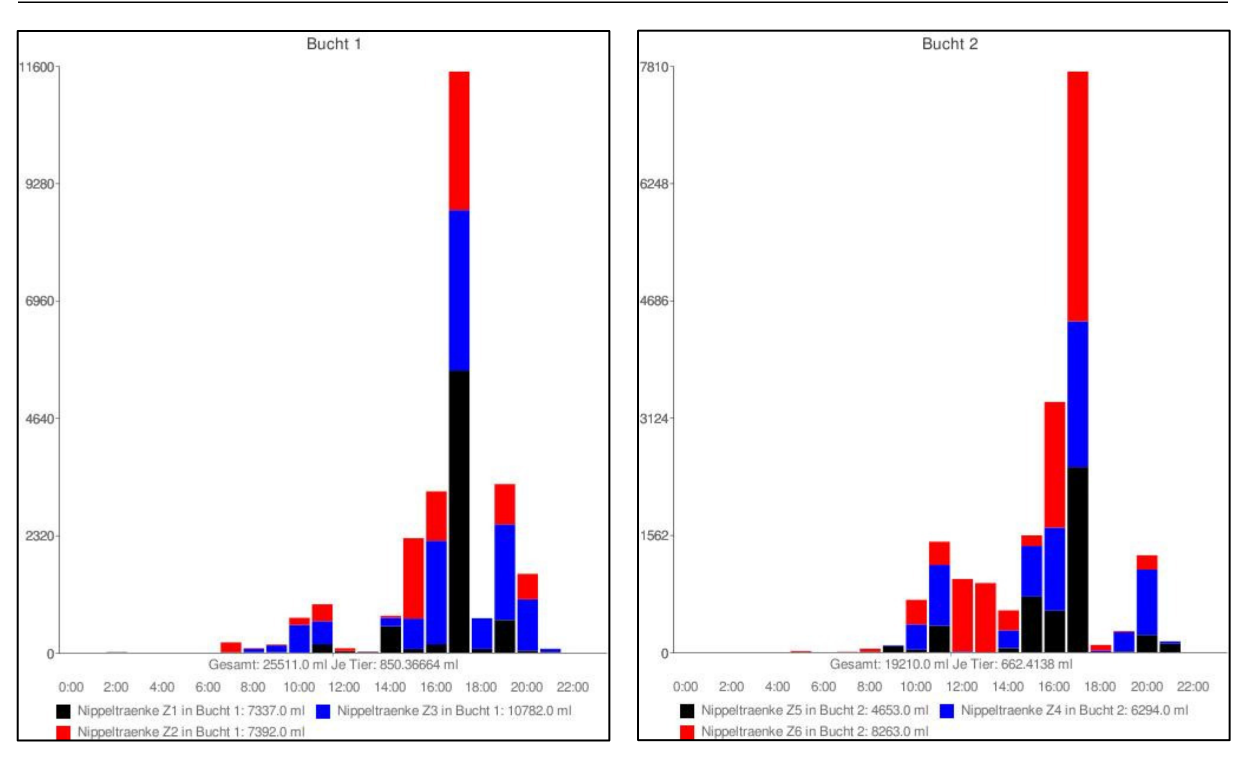


Abbildung 5.22: Darstellung von Verbrauchsmesserdaten

Mit dem Parameter mail_recipients werden die Adressaten der E-Mail definiert. Im obigen Beispiel ist die Mailingliste der einzige Adressat.

Da der Info Mailer Zugriff auf die Farming Cell Datenbank benötigt, sind die erforderlichen Angaben in der Datei MySQL_DB.properties hinterlegt.

MySQL_DB.properties

```
driver = com.mysql.jdbc.Driver
DB_SERVER = ***.***.***.***
DB_NAME = farmingcell
password = ********
user = farmingcell
url = jdbc:mysql://
```

Der Info Mailer besitzt keine grafische Benutzeroberfläche. Er wird in der Kommandozeile ausgeführt. Er akzeptiert zur Laufzeit keinerlei Tastatureingaben, eine Interaktion ist folglich nicht möglich. Wie Abbildung 5.23 zeigt, werden beim Start der Ausführungszeitpunkt, die zu erzeugenden Diagramme sowie die Empfänger der E-Mail genannt. Sobald mit der Erzeugung der E-Mail begonnen wird, werden Statusmeldungen ausgegeben.

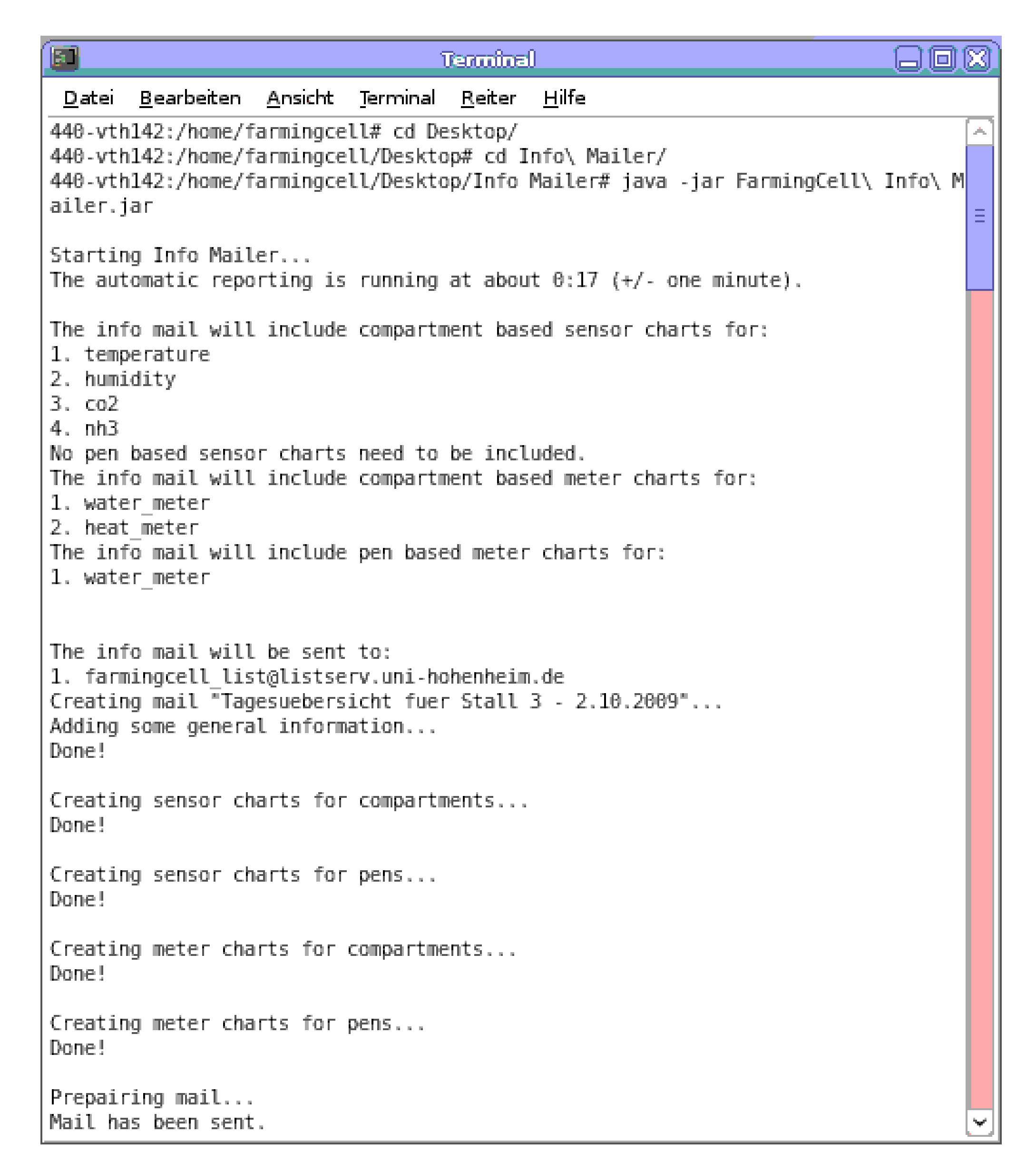


Abbildung 5.23: Kommandozeilenausgabe des Info Mailers

5.6.8 Webapplikation

Bei der Webapplikation[25] handelt es sich um eine mit der Java Server Pages (JSP) Technologie implementierte, browserbasierte Benutzerschnittstelle. Sie ist sowohl auf mobilen Endgeräten, als auch mit jedem im lokalen Netzwerk befindlichen Computer nutzbar. Einzige Voraussetzung auf Clientseite ist neben dem Netzzugang ein Browser. Als serverseitige Laufzeitumgebung ist ein Tomcat Webserver geeignet.

Die Webapplikation dient dem Zweck, die Dokumentation durchgeführter Managementaufgaben zu ermöglichen und bei deren Durchführung zu unterstützen.

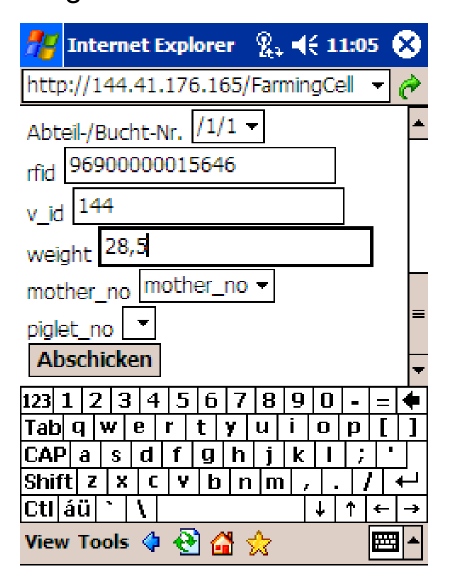
[25] www.farmingcell.de:8080/FarmingCell_Webapplikation

Sie arbeitet direkt auf dem Datenbestand der Farming Cell. Autorisierte Personen können die folgenden Tätigkeiten mithilfe der Webapplikation dokumentieren:

- Einstallen (Einzeltier oder Tiergruppe)
- Umstallen (Einzeltier oder Tiergruppe)
- Ausstallen (Einzeltier oder Tiergruppe)
- Diagnosestellung und Medikamentierung (Einzeltier)

Mithilfe eines mobilen Gerätes wie beispielsweise dem Handlesegerät WORKABOUT PRO C mit RFID Lesemodul AIR200 ist die Nutzung der Webapplikation im Stall möglich (Wireless LAN vorausgesetzt). Die RFID Leseeinrichtung des Handlesegerätes wird von der Webapplikation unterstützt, d.h. Tiere sind berührungslos anhand ihrer Ohrmarke identifizierbar (vgl. Abbildung 3.8).

Die folgende Abbildung 5.24 zeigt die Benutzeroberfläche auf dem Handlesegerät für den Einstallprozess und bei der Eingabe von Behandlungsdaten. Das Nachtragen von Daten unter Nutzung eines Computers zu einem späteren Zeitpunkt ist ebenso möglich.



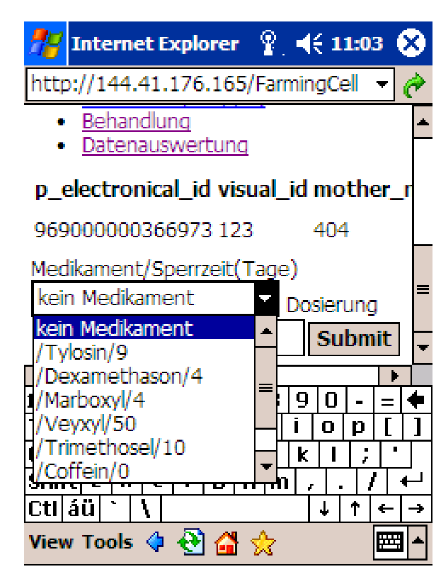


Abbildung 5.24: Webapplikation - Bildschirmmasken
Links: Einstallen; Rechts: Medikamentierung

Über die genannten Managementtätigkeiten hinaus bietet die Webapplikation die Möglichkeit, Dateien im Microsoft Excel Format zu importieren, welche Tierdaten zum Inhalt haben. Die Dateien sind Exporte der Software Supersau der Firma CLAAS

Agrosystems GmbH & Co. KG (ehemals AGROCOM GmbH & Co. Agrarsystem KG) und enthalten Informationen über potentielle Masttiere (vgl. Tabelle 5.31).

Tabelle 5.31: Beispiel Importdatei

Name	Nr	Geb-Dat	Geschlecht	Rasse	Name
344	90	28.11.2008	1	PI*DL	Momo
402	78	21.12.2008	1	PI*DL	Brutus
402	79	21.12.2008	1	PI*DL	Brutus
404	80	14.03.2009	1	PI*DL	Basti
404	81	14.03.2009	1	PI*DL	Basti

Name: Nummer der Muttersau   Nr: Ferkelnummer
Geschlecht: 1 – weiblich, 2 männlich   Name: Name des Ebers

Für den erfolgreichen Import ist es notwendig, die Struktur von Tabelle 5.31 einzuhalten. Ein Beispiel für eine korrekt formatierte Datei ist im Projekt Webapplikation auf dem VisualSVN Server zu finden.

Der Export-Import Vorgang sollte vor dem Einstallen durchgeführt werden, damit der aktuelle Ferkelbestand im Farming Cell System vorliegt und während des Einstallprozesses lediglich die Zuordnung der elektronischen Ohrmarkennummer und des Einstallgewichtes erfolgen muss. Über die Zeitersparnis während des Einstallens hinaus bietet das Vorgehen den Vorteil, zusätzliche Informationen eines jeden Ferkels aus der Software Supersau übernehmen zu können. Dazu zählen beispielsweise die Abstammung, das Geschlecht und das Geburtsdatum.

5.6.9 Datenexport

Insbesondere im Hinblick auf wissenschaftliche Auswertungen ist es notwendig, Daten der Farming Cell Datenbank exportieren zu können, um sie anschließend weiterzuverarbeiten. Dieser Anforderung wird auf zwei verschiedene Weisen Rechnung getragen. Zum einen wurde eine Applikation entwickelt, die, aufsetzend auf einem Reporting Framework, die Anzeige und den Export ausgewählter Daten ermöglicht. Zum anderen ist der Datenbankzugriff mithilfe der zur

Softwarezusammenstellung XAMPP (vgl. Kapitel 4.2) gehörenden Webapplikation phpMyAdmin möglich. Beide Möglichkeiten stellt dieses Kapitel vor.

5.6.9.1 Reporting Applikation

Ein für den Endnutzer komfortabler Weg, ausgewählte Daten der Farming Cell anzuzeigen und zu exportieren, ist die Verwendung eines entwickelten Reporting Tools. Es nutzt das Framework BIRT[26] und ist webbasiert[27] verfügbar, kann alternativ jedoch ebenso in Eclipse (vgl. Kapitel 4.2) ausgeführt werden. Das Reporting Tool kann nicht mit mobilen Geräten genutzt werden.

Wie Abbildung 5.25 zeigt, kann der Anwender Einfluss nehmen auf die Art der anzuzeigenden Daten, den Ort bzw. die Tiergruppe für den die Werte erhoben wurden und den Betrachtungszeitraum.

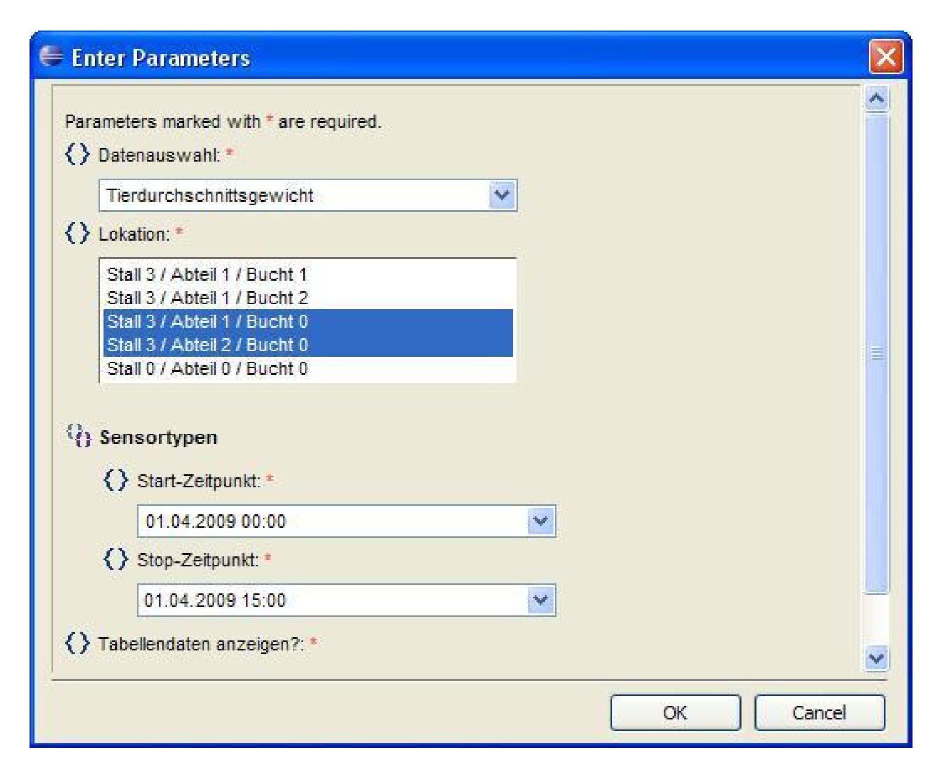


Abbildung 5.25: Reporting Applikation

[26] Business Intelligence and Reporting Tools (BIRT); Internetpräsenz: http://www.eclipse.org/birt/phoenix/
[27] Internetpräsenz: http://www.farmingcell.de:8080/birt-viewer/frameset?__report=all_views.rptdesign

Die folgende Datenauswahl steht zur Verfügung:

- Temperatur (Einzelwerte und Stundenmittel)
- Feuchtigkeit (Einzelwerte und Stundenmittel)
- CO_2 (Einzelwerte und Stundenmittel)
- NH_3 (Einzelwerte und Stundenmittel)
- CH_4 (Einzelwerte und Stundenmittel)
- N_2O (Einzelwerte und Stundenmittel)
- Helligkeit (Einzelwerte und Stundenmittel)
- Unterdruck (Einzelwerte und Stundenmittel)
- Lüftungsdurchsatz (Einzelwerte und Stundenmittel)
- Windstärke (Einzelwerte und Stundenmittel)
- Windrichtung (Einzelwerte und Stundenmittel)
- Luftdruck (Einzelwerte und Stundenmittel)
- Regen Enthalpie (Einzelwerte und Stundenmittel)
- Strommesser (Einzelwerte und Stundensummen)
- Wassermesser (Einzelwerte und Stundensummen)
- Wärmemesser (Einzelwerte und Stundensummen)
- Gasmesser (Einzelwerte und Stundensummen)
- Gruppengewicht
- Tierdurchschnittsgewicht (auf Gruppenbasis)
- Tageszunahme (auf Gruppenbasis)

Die Liste wird auf Basis der in der Farming Cell Datenbank vorhandenen Views erstellt. Sie ist durch zusätzliche Views erweiterbar, Änderungen am Quellcode der Applikation sind nicht erforderlich.

Ein Beispiel eines Reports zeigt Abbildung 5.26. Sollten mehrere Datenreihen für die gewählte Parameterkombination vorhanden sein, werden entsprechend viele Kurven angezeigt.
Generierte Darstellungen können gespeichert werden (beispielsweise im Format PDF). Die der Darstellung zugrunde liegenden Daten sind exportierbar. Mögliche Formate sind zum Beispiel XLS und CSV.

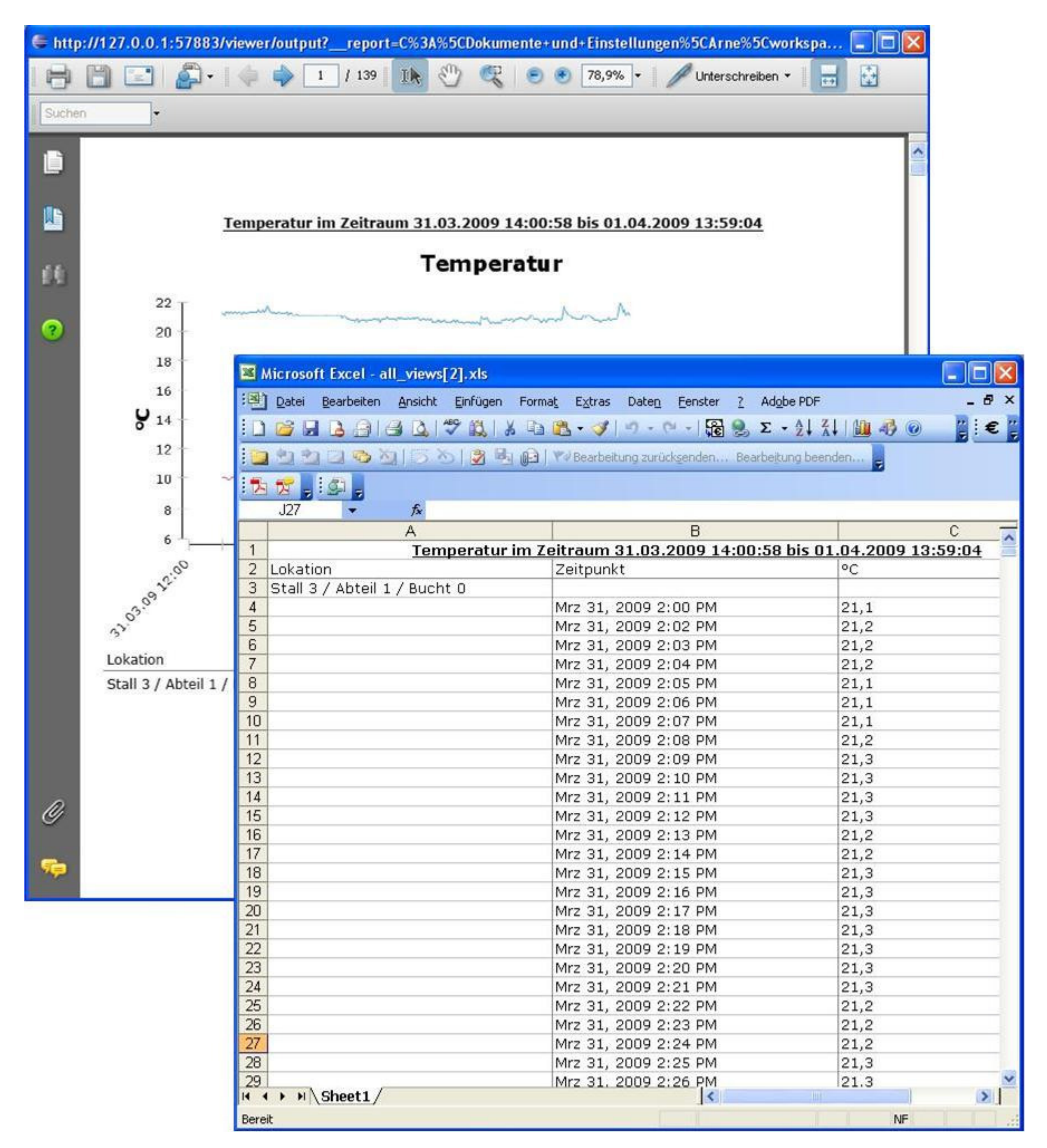


Abbildung 5.26: Report als PDF und Export in Excel

5.6.9.2 phpMyAdmin

Bei phpMyAdmin[28] handelt es sich um eine Webapplikation zur Administration von MySQL Datenbanken. Sie ist Bestandteil der Softwarezusammenstellung XAMPP (vgl. Kapitel 4.2). Über die Möglichkeit hinaus, das Datenbankschema zu bearbeiten, ist auch der Zugriff auf ausgewählte Inhalte möglich. Diese können in unterschiedliche Zielformate exportiert werden.

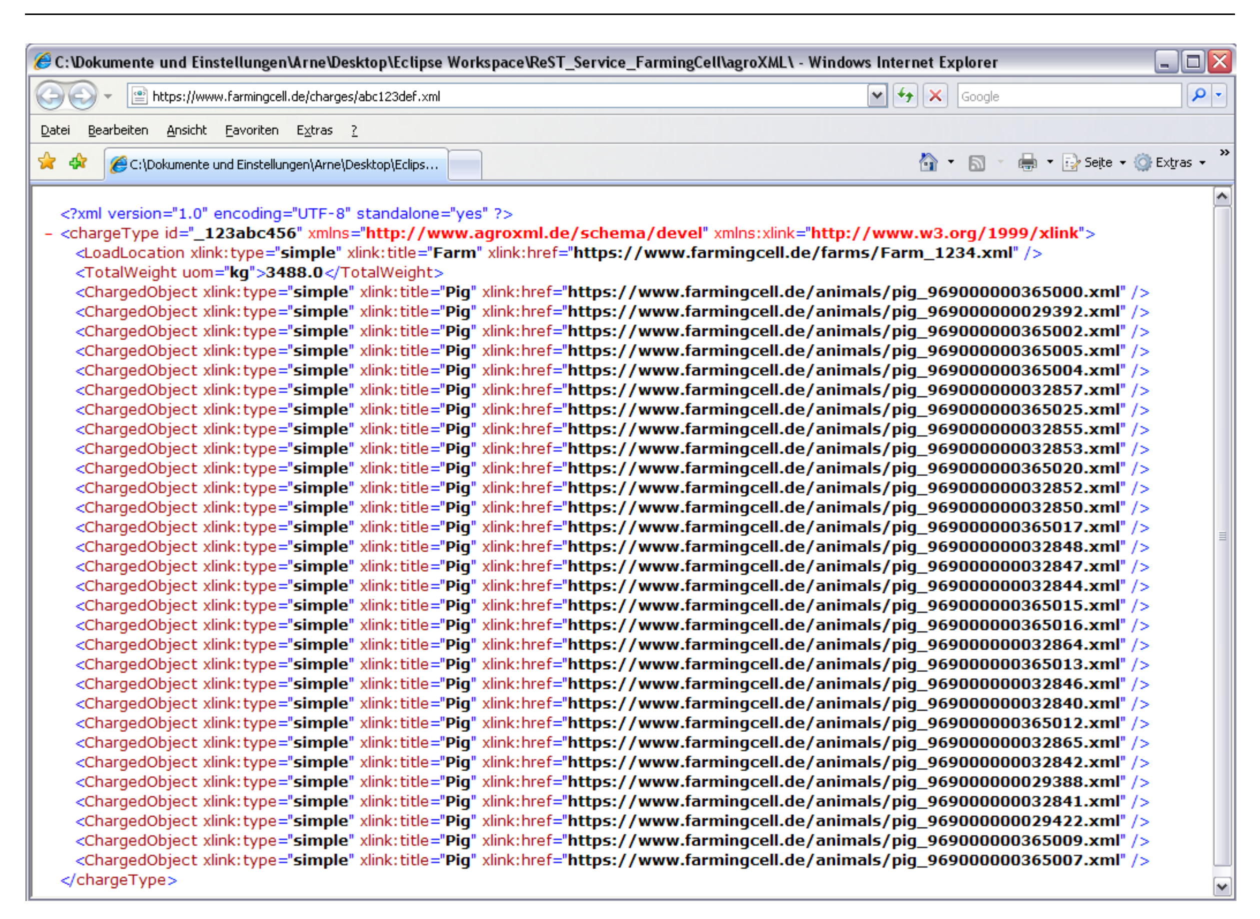
[28] Internetpräsenz: http://www.phpmyadmin.net

5.6.10 REST Service

REST bezeichnet einen Softwarearchitekturstil für verteilt arbeitende Informationssysteme wie das World Wide Web. Mit Hilfe dieser Technik können Informationen aus weltweit gestreuten Datenbeständen verknüpft und so zum Beispiel für die Verfolgung von Tieren genutzt werden (HERD et al., 2007).

Ein REST Webservice, wie er für die Farming Cell entwickelt wurde, setzt auf bewährte Protokolle wie HyperText Transfer Protocol Secure (HTTPS) und referenziert auf jede Ressource mit einem eindeutigen Uniform Resource Identifier (URI).
Im Falle der Farming Cell liegen die bereitgestellten Informationen im agroXML Format vor. Das in Kapitel 5.1.2 vorgestellte Framework ermöglicht das Erzeugen der notwendigen agroXML Instanzen.
Ebenso, wie sich Anfragende am REST Server mit Benutzernamen und Passwort anmelden müssen um Daten zu erhalten, weist sich der Server im Gegenzug mit einem Zertifikat aus, welches die Echtheit des Servers versichern soll.

Der Aufruf des in Abbildung 5.27 gezeigten Beispiels ist, eine Zugangsberechtigung vorausgesetzt, mithilfe des URI https://www.farmingcell.de/charges/abc123def.xml möglich.




```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <chargeType id="_123abc456" xmlns="http://www.agroxml.de/schema/devel" xmlns:xlink="http://www.w3.org/1999/xlink">
    <LoadLocation xlink:type="simple" xlink:title="Farm" xlink:href="https://www.farmingcell.de/farms/Farm_1234.xml" />
    <TotalWeight uom="kg">3488.0</TotalWeight>
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365000.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000029392.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365002.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365005.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365004.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032857.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365025.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032855.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032853.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365020.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032852.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032850.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365017.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032848.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032847.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032844.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365015.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365016.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032864.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365013.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032846.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032840.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365012.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032865.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032842.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000029388.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000032841.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000029422.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365009.xml" />
    <ChargedObject xlink:type="simple" xlink:title="Pig" xlink:href="https://www.farmingcell.de/animals/pig_969000000365007.xml" />
  </chargeType>
```


Abbildung 5.27: agroXML – Dokument einer Charge

Das implementierte Beispiel umfasst Tierchargen und damit verknüpfte Informationen zu landwirtschaftlichen Betrieben und Einzeltieren. Dokumente, wie das obige der Charge, können URI enthalten, welche auf weitere Dokumente, hier einen landwirtschaftlichen Betrieb und mehrere Einzeltiere, verweisen.

Die Dokumente liegen in Dateiform vor. Der mittels REST Schnittstelle abfragbare Datenbestand der Farming Cell kann daher erweitert werden, indem weitere Dateien in das entsprechende Verzeichnis der Applikation gelegt werden. Ferner wird durch das Vorhalten der Informationen in persistenter Form die Mehrfachausführung wiederkehrender Datenbankabfragen vermieden.

Der Quellcode der Applikation namens REST Service ist im SVN Repository verfügbar (vgl. Kapitel 5.4.3).

5.7 Zeitliche Taktung der Services

Da einige Softwarekomponenten automatisch operieren und zeitliche Abhängigkeiten zu anderen bestehen, ist eine korrekte Taktung dieser Services für den reibungslosen Betrieb notwendig.

Folgende Rahmenbedingungen und Abhängigkeiten gelten für die einzelnen Software Services der Farming Cell.

Schauer Service

- Ausführung nach letzter Fütterung des Tages (vgl. Fütterungszeiten unten).
- Ausführung nach TruTest Service, da dieser, sollte am gleichen Tag gewogen worden sein, die neue Tieranzahl und das neue Tierdurchschnittsgewicht in die Fütterung einträgt.

Fütterungszeiten (gültig während des gesamten Projektzeitraumes).

- 6:00 und 6:20
- 8:45 und 9:20
- 12:00 und 12:20
- 15:00 und 15:20
- 18:30 und 18:50
- 21:40 und 22:00

TruTest Service

- Ausführung nach einer möglichen Wiegung, jedoch am gleichen Tag.

Info Mailer

- Ausführung zwingend am Folgetag, da alle Daten des Vortages genutzt werden.

Ausgehend von den genannten Rahmenbedingungen ergibt sich die **Ausführungsreihenfolge** TruTest Service, Schauer Service und im Anschluss Info Mailer. Als Zeitfenster für die Ausführung der beiden erstgenannten ist das Zeitfenster 22:00 bis 23:59 sinnvoll, da es genug Zeit für die Abarbeitung der Aufgaben bietet und Arbeiten im Stall und damit einhergehende relevante Änderungen zu später Stunde nicht zu erwarten sind.
Da der Info Mailer ausschließlich auf Vortagesdatensätze zugreift, ist dessen Ausführung außerhalb der Arbeitszeit sinnvoll. Um die E-Mail mit den Vortagesinformationen bei Dienstbeginn am Morgen vorliegen zu haben, wurde als Ausführungszeitpunkt 00:05 gewählt.

Das Generieren der E-Mail Inhalte kann, abhängig von der gewählten Konfiguration (vgl. Kapitel 5.6.7) und dem Volumen der Daten in der Datenbank, mehrere Stunden in Anspruch nehmen.

6 Anwendungsbeispiele: Datenauswertung und -interpretation

Die Fülle an erfassten Daten aus den Bereichen Ressourcen, Klima und Einzeltiere eröffnet vielfältige Möglichkeiten der Auswertung. Neben der Darstellung einzelner Parameter, wie beispielsweise im Rahmen der täglich generierten Informations-E-Mail, ist die kombinierte Betrachtung mehrerer Parameter sehr interessant. Dieses Kapitel stellt eine Auswahl möglicher Betrachtungen anhand ausgewählter Beispiele dar. Als Datengrundlage dienen die Daten von Mastdurchgang 4 (vgl. Tabelle 6.1).

Tabelle 6.1: Mastdurchgänge

Durchgang	Einstallung	Ausstallung	Abteil/ Bucht	Anzahl Tiere
1	03.03.2008	Juni 2008	1 / 1und 2	60
2	19.08.2008	Dezember 2008	1 / 1und 2	60
3	03.03.2009	Juni 2009	1 / 1und 2	60
4	22.06.2009	Oktober/ November 2009	1 / 1und 2	60

1. Betrachtung: Futterverwertung

Die Futterverwertung ist ein wichtiger Indikator für die Effektivität des Mastprozesses. Sie beschreibt das Verhältnis von gefütterter Trockenmasse zu Gewichtszunahme.

Für die Berechnung der Futterverwertung wurden folgende Daten herangezogen:

1. Zusammensetzung der Futtermischung
2. Einzeltiergewichte von fünf Wiegetagen
3. Gruppenweise Futterverbräuche

Futtermischung

Für die Berechnung der Futterverwertung ist die Errechnung des Trockensubstanzanteils der Futtermischung erforderlich. Der Wasseranteil der Futtermischung beträgt 71,4%.

Tabelle 6.2: Futtermischung

	Anteil	Trockensubstanz	Eiweiß g/kg	Lysin g/kg	Cystin g/kg
Mischung	28,6%	88%	147	10,10	6,10

Anteil Trockensubstanz je kg Futtermischung (basierend auf Tabelle 6.2):
1 * 0,286 * 0,88 = 0,25168 = 25,168%

Einzeltiergewichte

Während des Mastdurchganges sind 5 Wiegungen erfolgt (vgl. Tabelle 6.3).

Tabelle 6.3: Gewichtsentwicklung

Datum (Masttag)	Tieranzahl		Tierdurchschnittsgewicht [kg]	
	Gruppe 1	Gruppe 2	Gruppe 1	Gruppe 2
22.06.2009 (0)	30	30	34,85	30,12
16.07.2009 (24)	30	29	49,17	42,02
29.07.2009 (37)	30	29	56,87	52,71
03.09.2009 (73)	29	29	94,26	87,23
24.09.2009 (94)	28	29	109,69	99,85

Die durchschnittliche Zunahme über 94 Tage beträgt somit 74,84 kg für die Tiere der Gruppe 1 und 69,73 kg für die Tiere der Gruppe 2. Dies entspricht durchschnittlichen Einzeltiertageszunahmen von 796,17 g und 741,81 g. Diese Werte sind hoch, vergleicht man sie mit dem durch den ZDS ermittelten Vergleichswert von 735 g (vgl. Anonymus, 2007). ZALUDIK (2002) hat im Rahmen ihrer Untersuchungen Tageszunahmen von 759 g ermittelt.

Futterverbräuche

Folgende Tabelle stellt die zwischen den Wiegungen ausgeteilten Futtermengen sowie deren Entsprechung in Trockenmasse dar.

Tabelle 6.4: Futterverbräuche

Zeitraum (Anzahl Tage)	**Gesamtfutter-verbrauch je Tier [kg]**		**Gesamtverbrauch Trockenmasse je Tier [kg]**[1]		**Tagesdurch-schnittsverbrauch Trockenmasse je Tier [kg]**	
	Gruppe 1	Gruppe 2	Gruppe 1	Gruppe 2	Gruppe 1	Gruppe 2
22.06.2009 – 15.07.2009 (24)	178,367	162,748	44,891	40,960	1,870	1,707
16.07.2009 – 28.07.2009 (13)	133,367	123,448	33,566	31,069	2,582	2,390
29.07.2009 – 02.09.2009 (35)	419,430	397,724	105,562	100,099	3,016	2,860
03.09.2009 – 23.09.2009 (21)	229,341	220,995	57,721	55,620	2,749	2,649

[1]Einen Trockensubstanzanteil von 25,168% zugrunde legend (s.o.)

Werden der Trockenmasseverbrauch und die Gewichtszunahme eines Zeitraumes ins Verhältnis gesetzt, lässt sich eine Aussage über die Futterverwertung treffen (vgl. Tabelle 6.5).

Tabelle 6.5: Futterverwertung

Zeitraum (Anzahl Tage)	**Durchschnittliche Gewichtszunahme je Tier [kg]**		**Gesamtverbrauch Trockenmasse je Tier [kg]**		**Trockenmasse je kg Zunahme [kg]**	
	Gruppe 1	Gruppe 2	Gruppe 1	Gruppe 2	Gruppe 1	Gruppe 2
22.06.2009 – 15.07.2009 (24)	14,32	11,9	44,891	40,960	3,135	3,442
16.07.2009 – 28.07.2009 (13)	7,7	10,69	33,566	31,069	4,359	2,906
29.07.2009 – 02.09.2009 (35)	37,39	34,52	105,562	100,099	2,823	2,900
03.09.2009 – 23.09.2009 (21)	15,43	12,62	57,721	55,620	3,741	4,407
					∅ 3,230	**∅ 3,266**

Der gemäß dem ZDS (vgl. Anonymus, 2007) in der Praxis übliche durchschnittliche Futtereinsatz von 2,92 kg je kg Zuwachs ist besser, als der für die Gruppen im Versuchsbetrieb ermittelte. Es ist auffällig, dass die Futterverwertung während des Mastdurchgangs stark schwankte. Als ein Grund hierfür werden technische Ursachen (Defekte, Programmfehler) der Fütterung vermutet. Die Einbrüche der Futterverbrauchskurve sind in Abbildung 6.1 erkennbar.

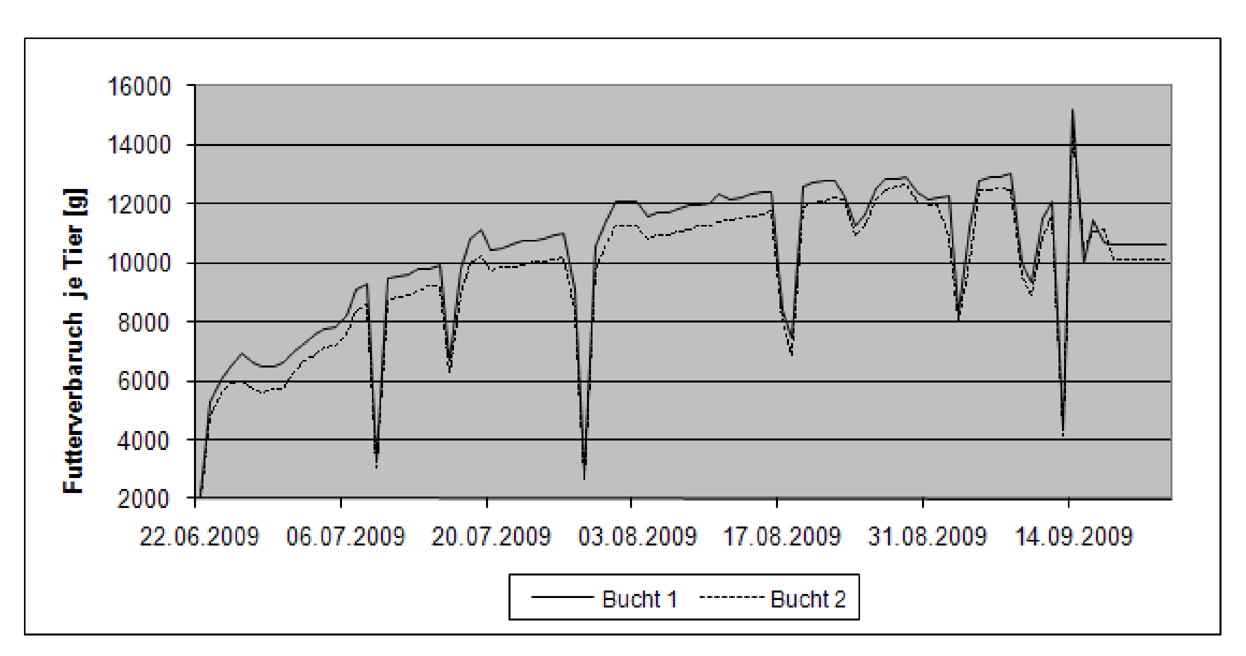


Abbildung 6.1: Futterverbrauch

2. Betrachtung: Wasseraufnahme in Abhängigkeit der Temperatur

Auf Basis der Mastgruppe erfolgt die Erfassung der Parameter Tränkewasseraufnahme und Futterverbrauch. Im Falle des erfassten Futterverbrauches pro Bucht und Tag ist anhand des bekannten Mischungsverhältnisses und des Trockensubstanzgehaltes die Errechnung der über das Futter aufgenommenen Wassermenge möglich. Die täglich von der Gruppe in einer Bucht über die Tränkenippel aufgenommene Wassermenge wird anhand der individuell erfassten Abgabemengen der Tränkenippel errechnet.

Die sich ergebenden Tierdurchschnittsverbräuche von Bucht 1 in Abteil 1 für den Zeitraum 07.08.2009 bis 03.09.2009 stellt die folgende Abbildung dar. Die Anzahl der in diesem Zeitraum in der Bucht befindlichen Tiere belief sich bis zum 10.08.2009 auf 30, danach auf 29. Das durchschnittliche Tiergewicht stieg im betrachteten Zeitraum von 65,42 kg auf 94,26 kg an.

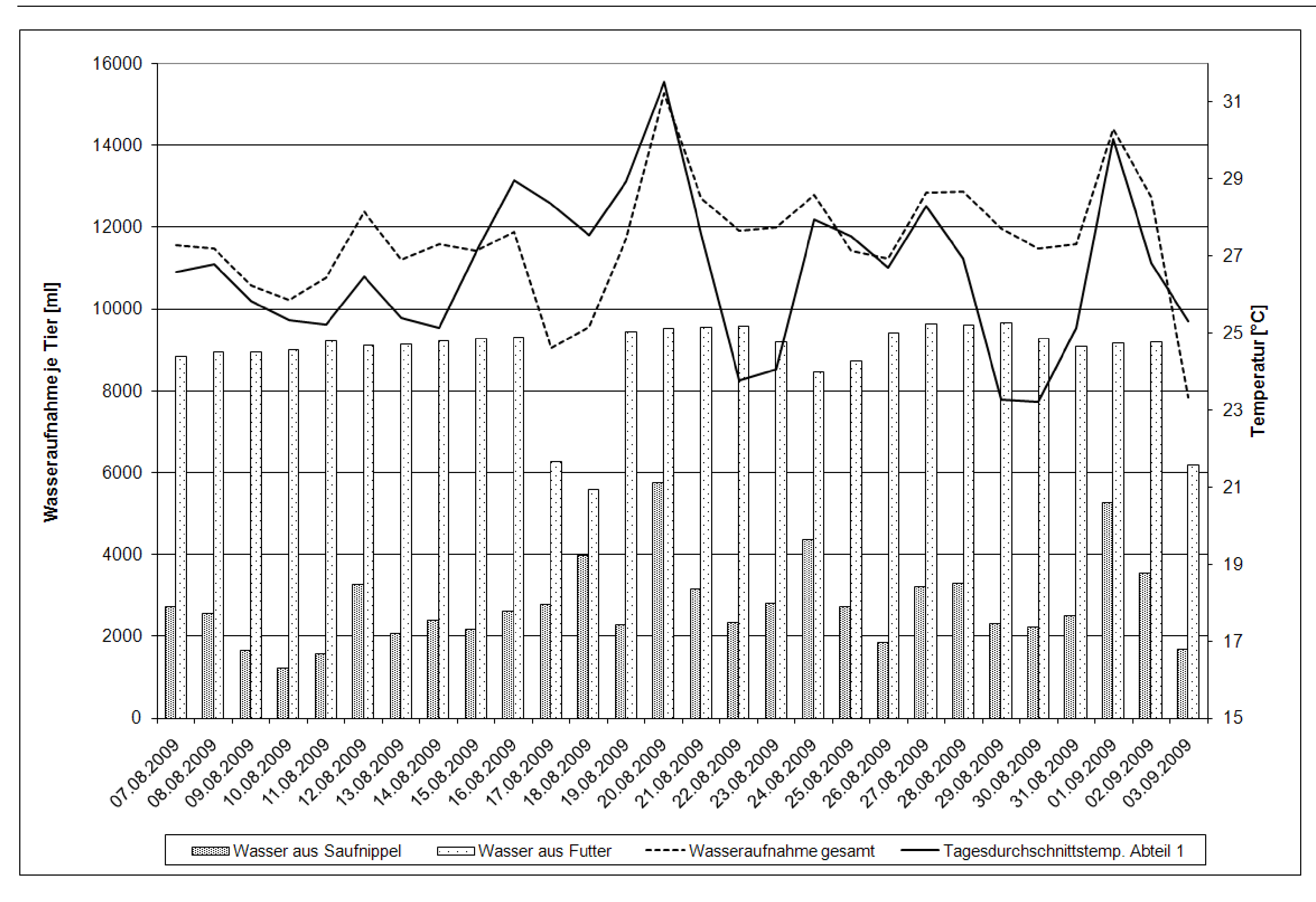


Abbildung 6.2: Tägliche Wasseraufnahme sowie Temperaturwerte

Im betrachteten Zeitraum kam es zu einer hohen Tagesdurchschnittstemperaturschwankung von 8,4 °C (min. 23,1, max. 31,5 °C). Die Tagesdurchschnittstemperatur im Stall lag mit durchschnittlich 25,31 °C über der durch HEINRITZI et al. (2006) ermittelten Optimaltemperatur von 20 °C.

Der Temperaturverlauf spiegelt sich in der aufgenommenen Tränkewassermenge wieder, wohingegen die Menge des über das Futter aufgenommenen Wassers relativ konstant ist. Der Grund für die Einbrüche des Futterverbrauchs an den Tagen 17. und 18.08.2009 ist unbekannt. Es werden technische Ursachen bei der Fütterungsanlage vermutet. Die verminderte Futteraufnahme am 03.09.2009 wird auf die an diesem Tag stattgefundene Wiegung und den resultierenden Stress zurückgeführt.
Ein Zusammenhang zwischen der aufgenommenen Wassermenge und dem Tiergewicht von anfangs durchschnittlich 65,42 kg und später 94,26 kg ist nicht zu erkennen. Der nach BÜSCHER et al. (2008) mit dem Gewichtszuwachs einhergehende Anstieg der Wasseraufnahme (vgl. Tabelle 6.6) konnte nicht beobachtet werden.
Die durchschnittliche Tagesmenge des über die Tränke abgegebenen Wassers beträgt für den Zeitraum 07.08.2009 bis 03.09.2009 2.788 ml je Tier. Dieser Wert überschreitet den täglichen Tränkewasserbedarf eines Mastschweins nach KTBL (2006) von 1800 – 2500 ml, scheint jedoch plausibel angesichts der hohen Temperaturen in diesem Zeitraum.
Die durchschnittliche Wassermenge, die über das Futter aufgenommen wird, beläuft sich auf 8.870 ml pro Tier und Tag. Der tägliche Gesamtwasserverbrauch beträgt demnach 11.658 ml und liegt über dem von RUDOVSKY (2008) angegebenen Wasserbedarfswert von 8.500 – 11.000 ml (vgl. Tabelle 6.6).

Tabelle 6.6: Wasserbedarf von Mastschweinen (RUDOVSKY, 2008)

Lebendmasse [kg]	Wasserbedarf [l / Tier und Tag]
< 50	3 – 6
50 – 80	5 – 8,5
80 – 120	8,5 – 11

Neben den hohen Temperaturen im Auswertungszeitraum, wird auch das Trinkverhalten als mögliche Ursache für die hohen Werte in Betracht gezogen. Die Tiere trinken in kurzen Intervallen und spielen mit den Tränkenippeln. Beides führt zu stoßartigen Wasserabgaben, was ein Nachlaufen der Flügelräder des Wasservolumenmessers und somit Fehlmessungen begünstigt. Es wurde versucht, dieser Tatsache durch die Ermittlung und den Einbezug von Korrekturfaktoren zu begegnen (vgl. Tabelle 5.12).

Zusammenfassend ist festzuhalten, dass die Tiere eine relativ konstante Menge Wasser über das Futter beziehen, durchschnittlich ca. 74% der Tagesmenge, und den temperaturbedingten Mehrbedarf über die Tränken ausgleichen.

3. Betrachtung: Gewichtszunahme in Abhängigkeit der eberseitigen Abstammung

Der Datenbestand eines jeden Masttieres in der Farming Cell umfasst unter anderem seine Abstammung und die individuellen Gewichte zu verschiedenen Zeitpunkten. Somit ist es möglich, die Tageszunahmen der Einzeltiere zu errechnen und zu Gruppen aggregiert darzustellen. Als Aggregationsparameter wurde die eberseitige Abstammung gewählt. Es wurden nur Eber einbezogen, deren Anzahl Nachkommen im Datenbestand ≥ 4 war. Der betrachtete Zeitraum reichte vom 22.06.2009 bis zum 24.09.2009 und deckt den Mastdurchgang nahezu vollständig ab (vgl. Tabelle 6.3).

Tabelle 6.7: Tageszunahmen in Abhängigkeit der eberseitigen Abstammung

Eber	Anzahl Nachkommen in Mastgruppe	Tageszunahmen der Nachkommen [kg]		
		arithmetisches Mittel	min.	max.
1	6	0,80	0,73	0,90
2	4	0,75	0,63	0,88
3	8	0,78	0,55	0,94
4	13	0,75	0,57	0,93
5	4	0,87	0,75	0,99
6	6	0,74	0,70	0,84
7	4	0,86	0,79	0,94
Gesamt	45	0,78	0,55	0,99

Die errechneten mittleren Tageszunahmen jeder Gruppe sind in Tabelle 6.7 dargestellt und werden in der folgenden Abbildung 6.3 zusammen mit den zugrunde liegenden durchschnittlichen Zunahmen der Einzeltiere visualisiert

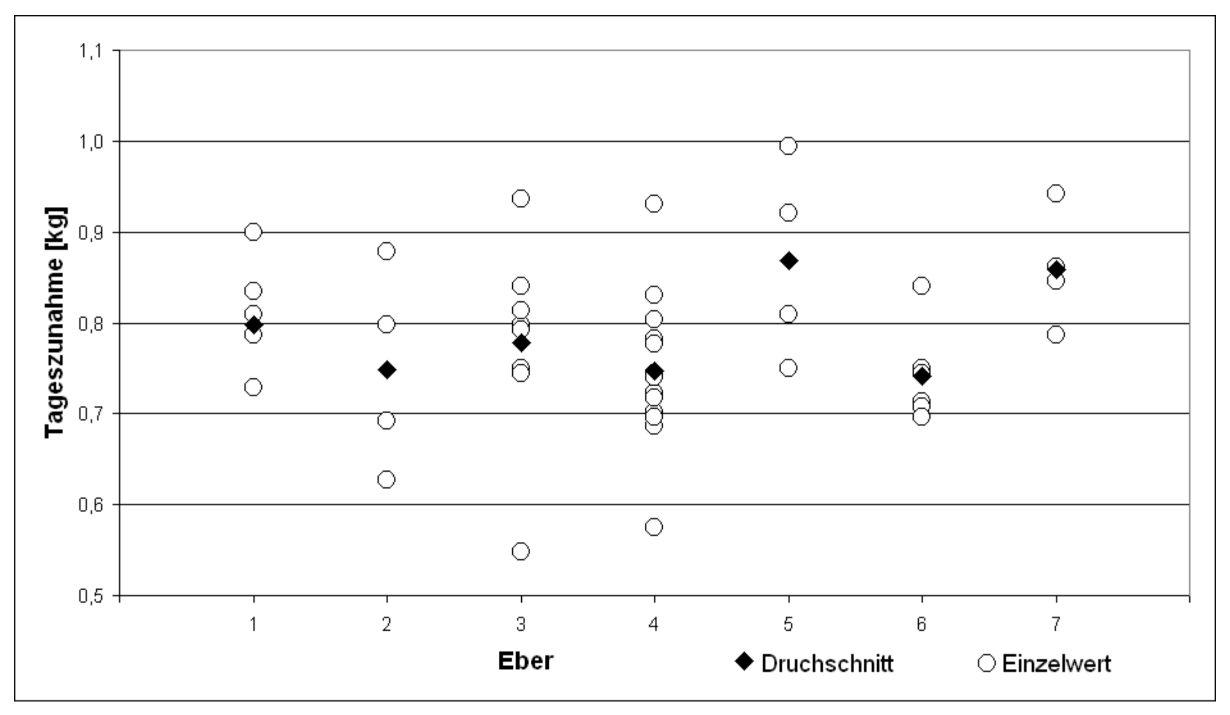


Abbildung 6.3: Durchschnittliche Tageszunahmen der Einzeltiere in Abhängigkeit der eberseitigen Abstammung

Die Abbildung verdeutlicht die vorhandene Streuung der durchschnittlichen Zunahmen der Einzeltiere von 550 g bis 990 g am Tag und lässt einen eberseitigen Einfluss vermuten.
Gründe für die niedrigen Tageszunahmen einzelner Nachkommen der Eber 3 und 4 konnten nicht ausgemacht werden. Aufgrund der geringen Stichprobengröße wurde auf eine statistische Auswertung verzichtet.

4. Betrachtung: NH_3 und CO_2 Konzentrationen in Abteil 1 an einem Tag mit Wiegeereignis

Die Möglichkeit der minütlichen Datenerfassung gestattet eine detaillierte Darstellung des Konzentrationsverlaufes von CO_2 und NH_3 in Abteil 1. Abbildung 6.4 zeigt einen solchen Verlauf am Beispiel des 03.09.2009. An diesem Tag erfolgte eine Wiegung der Tiere. Die durchschnittlichen Tiergewichte am dargestellten Tag betrugen 94,26 kg für Bucht 1 und 87,23 kg für Bucht 2 (vgl. Tabelle 6.3), die Tieranzahl in jeder der beiden Buchten war 29.

Bei der Interpretation der in Abbildung 6.4 dargestellten Daten ist zu beachten, dass der NH_3 Sensor eine zu geringe Empfindlichkeit zeigt, wodurch Konzentrationsschwankungen nicht im vollen Maße erfasst wurden. Der CO_2 Sensor lieferte zu hohe und hinsichtlich der Genauigkeit schwankende Werte (ADRION, 2009). Nichtsdestotrotz sind in der Darstellung Einflussfaktoren wie Futterausgaben (senkrechte Markierungen) und die zwischen 10:00 und 11:30 durchgeführte Wiegung deutlich zu erkennen. Die durch die Ereignisse ausgelöste Aktivität der Tiere lässt die Konzentration beider Gase sichtbar ansteigen, wie es auch BEA (2004) und GALLMANN (2003) beschreiben.

Ein an mehreren Tagen beobachtetes Phänomen ist ein CO_2 Peak im Zeitraum 22:10 – 22:20, welches bereits durch ADRION (2009) während seiner Messungen im leeren Abteil beobachtet wurde. Eine Ursache konnte nicht ausgemacht werden. Außeneinflüsse durch das zweite Abteil oder die angrenzende Biogasanlage schloss ADRION (2009) aus. Dies gilt es erneut zu prüfen.

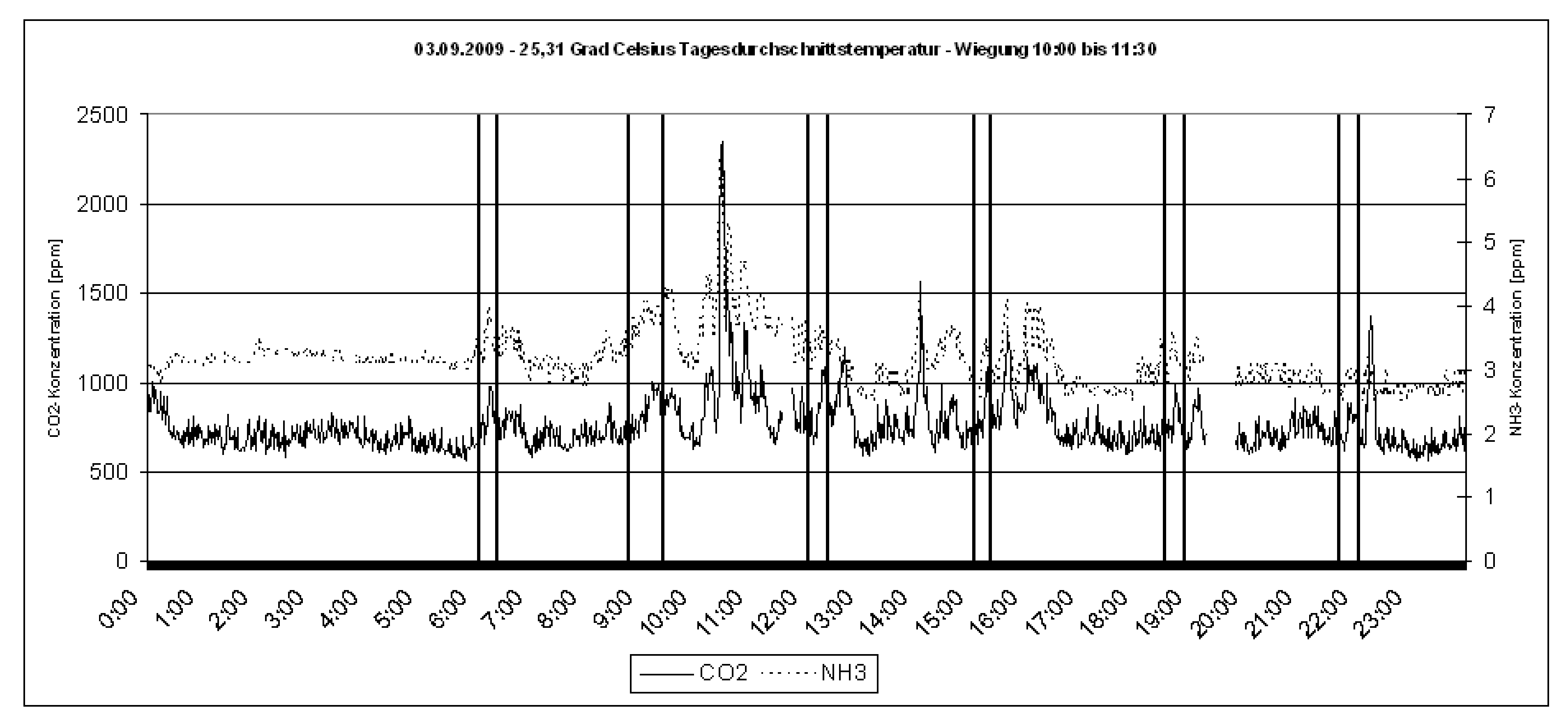


Abbildung 6.4: Tagesverlauf der CO2 und NH3 Konzentration

Tabelle 6.8: Fütterungszeiten

Austeilung 1	6:00	8:45	12:00	15:00	18:30	21:40
Austeilung 2	6:20	9:20	12:20	15:20	18:50	22:00

5. Betrachtung: Verlauf der CO_2 Konzentration in Abteil 1 im Zeitraum 10.08.2009 – 03.11.2009

Die Erfassung der Temperatur und der CO_2 Konzentration in Abteil 1 sowie der Vorraumtemperatur erfolgte minütlich. Die daraus errechneten Tagesmittelwerte sind zusammen mit der Anzahl Masttiere in Abteil 1 für den Zeitraum 10.08.2009 – 03.11.2009 in Abbildung 6.5 dargestellt.

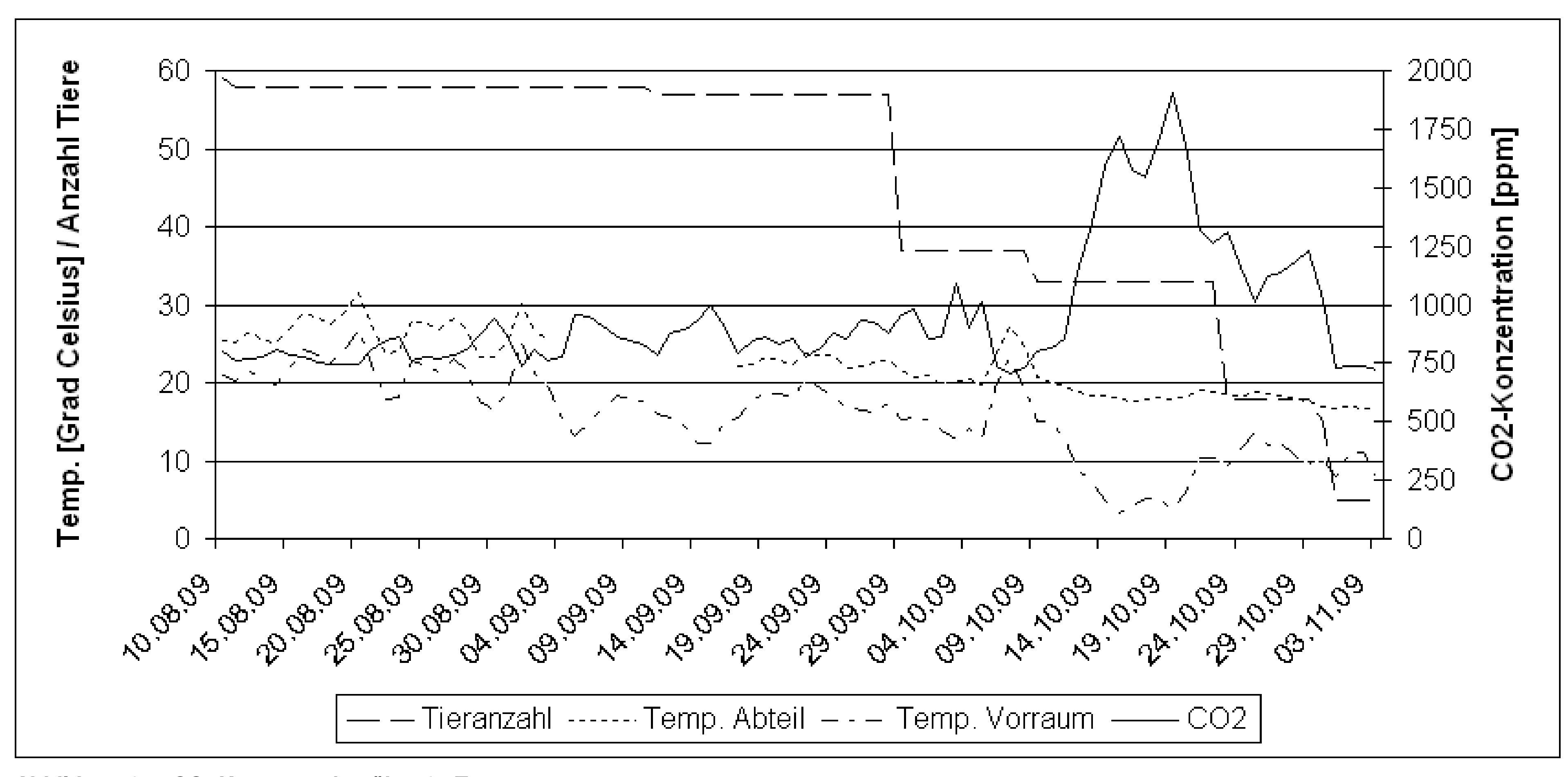


Abbildung 6.5: CO_2 Konzentration über 85 Tage

Der gewählte Zeitraum ist betrachtenswert, da sich diejenigen die CO_2 Konzentration beeinflussenden Parameter Vorraumtemperatur und Tieranzahl (GALLMANN, 2003) deutlich verändern. Es lassen sich mehrere Effekte beobachten.

Eine Abhängigkeit der CO_2 Konzentration von der Temperatur im Vorraum ist im ersten Abschnitt des Betrachtungszeitraumes deutlich sichtbar. Bis zum 02.09.2009 kann ein inverser Verlauf der beiden Kurven beobachtet werden. Der Verlauf der Innentemperatur folgt in diesem Abschnitt dem der Vorraumtemperatur, der Verlauf der CO_2 Konzentration ist gegenläufig. Ebendies gilt für den Zeitraum 17.09.2009 bis 08.10.2009.
Ab dem 09.10.2009 verläuft die Innentemperatur trotz Kälteeinbruch deutlich konstanter als im vorangegangenen Zeitraum[29].
Ebenso resultiert der Kälteeinbruch in der zweiten Oktoberhälfte in einem unmittelbaren Anstieg der CO_2 Konzentration. Die Ursache hierfür liegt in der Lüftungssteuerung begründet, die aufgrund der niedrigen Außentemperatur die Luftaustauschrate auf ein Minimum senkt, um die Innentemperatur im Soll Bereich zu halten. Die abnehmende Tieranzahl verstärkt diesen Effekt. Da die durch die Tiere abgegebene Wärme mit deren Anzahl abnimmt, ist das Temperaturniveau im Stall niedriger und weniger kühle Frischluft kann bis zum Erreichen der Mindesttemperatur zugeführt werden. Die Entkoppelung von Innen- und Außentemperaturverlauf ist auf denselben Grund zurückzuführen.
Die sich anschließenden wärmeren Tage sind anhand des leicht verzögerten Abfallens der CO_2 Konzentration deutlich erkennbar.

Zusammenfassend lässt sich festhalten, dass die CO_2 Konzentration im Stall indirekt von der Vorraumtemperatur und maßgeblich von der davon abhängigen Luftaustauschrate beeinflusst wird. Auch die Tieranzahl hat eine Auswirkung auf die CO_2 Konzentration im Stall. Die Beobachtungen decken sich mit den Aussagen von GALLMANN (2003).

[29] Eine Wärmezufuhr mittels Heizung erfolgte erst ab dem 28.10.2009, als die Tagesdurchschnitts-vorraumtemperatur auf ca. 10 °C fiel und lediglich noch 18 Tiere im Abteil waren.

7 Diskussion und Ausblick

Die Farming Cell ist ein prototypisches System. Sie war im produktiven Einsatz und lieferte Erkenntnisse über Anforderungen an derartige Systeme und Herausforderungen bei deren Umsetzung.

Dieses Kapitel beginnt mit einer Bewertung des Gesamtsystems, an welche sich Ausführungen zu Aspekten anschließen, die als verbesserungswürdig eingestuft werden. Den Abschluss des Kapitels bildet die Bewertung der gewählten Methoden und Technologien, die während des Entwicklungsprozesses zum Einsatz kamen.

7.1 Bewertung Gesamtsystem

Ein System zu schaffen, welches der Dokumentation des Produktionsprozesses dient und den Landwirt unterstützt, war die Motivation für den Aufbau die Farming Cell. Neben diesen beiden Aspekten, werden im Folgenden die Erweiterbarkeit und Übertragbarkeit des Systems und einzelner Komponenten, seine Zuverlässigkeit sowie seine Eignung als wissenschaftliches Werkzeug beleuchtet.

7.1.1 Prozessdokumentation und -unterstützung

Das Ziel, Prozesse zu dokumentieren und bei deren Durchführung zu unterstützen, kann nur mithilfe einer geeigneten Datengrundlage erreicht werden. Die Farming Cell bietet Möglichkeiten, um abgeschlossene Prozesse nachzuvollziehen und zu analysieren. Ebenso ist es möglich, den Prozess zu seiner Laufzeit zu beobachten, um gegebenenfalls Maßnahmen zu dessen Optimierung ergreifen zu können. Eine weitere Form der Prozessunterstützung ist die implementierte automatische Anlagenkommunikation.

Der theoretisch mögliche Nutzen ist in allen genannten Fällen direkt abhängig von der Art der erfassten Daten, der Frequenz ihrer Erfassung sowie ihrer zeitlichen Zuordnung zu Vorgängen, Orten oder Objekten. Alle genannten Parameter bestimmen den Grad der Prozessdokumentation und sind aus diesem Grund individuell festlegbar.

Das zugrunde liegende Datenmodell der Farming Cell ist in der Lage, alle Daten während und nach dem Prozess miteinander in Beziehung zu setzen. Dies ist die besondere Stärke des Systems. Auch rückwirkend ist das Verknüpfen verschiedener Informationen mithilfe des Parameters Zeit und dem Orts- bzw. Tierbezug möglich.

Die **Prozessdokumentation** ist folglich hinsichtlich aller derzeit erfassten Daten sichergestellt. Entsprechende Möglichkeiten wurden in dieser Arbeit vorgestellt und implementiert. Potential besteht hinsichtlich einer Erweiterung des Datenmodells um Informationen, die betriebsextern generiert werden. Insbesondere die Erfassung von Daten von unmittelbar vor- und nachgelagerten Wirtschaftspartnern, wie beispielsweise Futtermittellieferanten und Schlachthöfen, erscheint sinnvoll.

Hinsichtlich der **Prozessunterstützung** spielt die zeitnahe Bereitstellung von Daten die entscheidende Rolle. Insbesondere ist hierbei die Aufbereitung der Daten zu einer Information und die Verfügbarkeit für den Menschen bzw. die Weitergabe an Steuerungseinheiten maßgeblich.
Die Farming Cell gewährleistet mit ihrer Webapplikation jederzeit den Zugriff auf aktuelle Informationen und bietet Prozessbeteiligten eine Dokumentationsmöglichkeit für ausgewählte Arbeitsschritte. Da der Fokus des Projektes Farming Cell auf der Datenerfassung und -haltung lag, bietet die Benutzerschnittstelle lediglich grundlegende Funktionalität. Diese ist für den Schweinemastprozess hinreichend, Erweiterungpotential besteht dennoch.
Über die Webapplikation hinaus gestatten die vorgestellten Datenexportwerkzeuge den Zugriff auf alle Daten der Datenbank. Ferner ermöglichen Sie die Erstellung individueller Dokumentationen in unterschiedlichen Formaten. Wenngleich auf diese Weise eine Vielzahl unterschiedlicher Dokumentationen generierbar ist, besteht Potential für weitere, die es zu definieren gilt.
Die automatische Kommunikation zwischen Anlagen konnte aufgrund der aufwendigen Entwicklungsarbeit lediglich beispielhaft für die Fütterungsanlage, die Lüftungsanlage und die Tierwaage realisiert werden. Der Grad der Kommunikationsautomatisierung im System wird daher derzeit als unzureichend bewertet. Um diesbezüglich eine Verbesserung zu erzielen, sind Gespräche mit Anlagenherstellern, eine Analyse der Prozesse und die Ableitung des gewünschten Automatisierungsgrades voranzustellen.

7.1.2 Erweiterbarkeit, Übertragbarkeit

Die **Erweiterbarkeit** ist differenziert nach Datenquellen und Datensenken zu betrachten. Zu der Gruppe der Datenquellen gehört neben Anlagen, Sensoren, Verbrauchsmessern und Fremdsystemen auch der Mensch, welcher manuelle Eingaben tätigt. Die Gruppe der Datensenken umfasst Benutzer- und

Fremdsystemschnittstellen, Anlagen sowie die Daten verarbeitenden Dienste der Farming Cell.
Da der Aufwand für die Integration einer neuen Datenquelle oder –senke maßgeblich von deren Details abhängig ist, werden an dieser Stelle lediglich allgemeine Einschätzungen zu ausgewählten Datenquellen und -senken getroffen. Die folgenden Erläuterungen setzen eine ISOagriNET konforme Integration als Ziel voraus. Die Integration einer neuen Datenquelle kann die Erweiterung des Datenmodells erforderlich machen, sofern neue ISOagriNET Entitäten verwendet werden.

Die Gruppe nicht ISOagriNET konform kommunizierender **Anlagen** stellt die größte Herausforderung hinsichtlich einer Einbindung zum Zwecke des Datenaustausches dar. Zum einen sind ihre Kommunikationsprotokolle oftmals proprietär oder nur mit großem Aufwand nachzubildenden. Zum anderen verfügen sie in der Regel nicht über eine Ethernetschnittstelle. Erfolgt keine herstellerseitige Implementierung von ISOagriNET, ist der Aufwand einer Implementierung durch Dritte im Verhältnis zum Nutzen tendenziell zu groß.

Sowohl die Integration der **Sensoren**, als auch die der **Verbrauchsmesser** erfordert zusätzliche Hardware, sogenannte Gateways, welche ihre Messwerte im Netzwerk zugänglich machen. Kommen in der Farming Cell Sensoren mit bisher nicht unterstützten Schnittstellen zum Einsatz, sind andere Gateways erforderlich. Unterstützt ein neues Gateway ISOagriNET, ist es für den Einsatz in der Farming Cell geeignet.

Der Austausch von Daten zwischen **Fremdsystemen** und der Farming Cell ist gegenwärtig nicht ISOagriNET konform realisiert. Um eine Kommunikation zu erreichen, müssen die Farming Cell wie auch das Fremdsystem aufeinander abgestimmte Schnittstellen implementieren. Die Unterstützung des Fremdsystembetreibers vorausgesetzt, ist dies leicht möglich.

Eine **Übertragung** der Farming Cell als Gesamtsystem ist kaum möglich. Dies würde im Zielbetrieb eine identische Ausstattung hinsichtlich aller Anlagen und identische Prozessabläufe voraussetzen. Einzelne Komponenten, dies gilt insbesondere für die

Datenbank, sind hingegen portierbar. Differenziert zu betrachten sind die der Datenbank vorgelagerten Komponenten zur Datenerfassung, sowie die nachgelagerten zur Datennutzung. Sie sind zum Großteil nicht an den Prozess der Schweinmast gebunden, jedoch an die im Stall vorhandenen Anlagen und Messtechnik bzw. an Art und Umfang der in der Datenbank verfügbaren Informationen.

Die **Wiederverwendbarkeit** einzelner Komponenten oder ganzer Komponentengruppen ist insbesondere im Bereich der Messwerterfassung und -speicherung gegeben. Beispielsweise befindet sich die Hohenheimer Messwerterfassung (HME) zusammen mit dem ISOagriNET Parser und der Datenbank im Rahmen eines Projektes zur Bestimmung des Trocknungsverhaltens von Gärresten im Einsatz. Das System dient dabei in einem Trockenlager für Biomasse zur Klimadatenerfassung und -speicherung.
Werden einzelne Komponenten oder große Teile der Farming Cell übernommen, ist zu prüfen, ob das System fehlerfrei skaliert. Dem Umfang verarbeitbarer Daten sind Grenzen gesetzt. Gegebenenfalls sind einzelne Softwarekomponenten wie die Datenbank oder der ISOagriNET Parser, beziehungsweise Hardwarekomponenten wie Switches, nicht in der Lage, die Datenmengen zu bewältigen.

7.1.3 Zuverlässigkeit

Die Farming Cell war in ihrer größten Ausbaustufe mehrere Monate im Einsatz. Nahezu alle Komponenten arbeiteten fehlerfrei und erfüllten ihre Aufgabe.
Besonders hervorzuheben sind die Hardwarekomponenten **Hohenheimer Messwerterfassung** (HME) und **Ethernetbox**, welche in zwei- bzw. dreifacher Ausführung im Einsatz waren. Sie stellten die Erfassung nahezu aller Werte von Sensoren und Verbrauchsmessern sicher.
Die im Vorraum befindliche HME führte darüber hinaus die Kommunikation mit dem **Waagenterminal** durch, welche nicht immer störungsfrei verlief. Aufgrund der unzuverlässigen seriellen Kommunikation war eine benutzerseitige Überwachung des Datentransfers und gegebenenfalls ein erneutes Anstoßen erforderlich. Datenverluste waren jedoch nie die Folge.
Schwierigkeiten traten ebenso bei der **Fütterungsanlage** auf. Der Auslöser für das Fehlverhalten, welches sich in Form sporadisch ausbleibender Futterausteilungen äußerte, ist unklar. Ein Zusammenhang mit der zwischen Management PC und

Fütterungscomputer täglich erfolgten Kommunikation konnte nicht identifiziert werden. Die Ursache wird im Fütterungscomputer vermutet.

Der Ausfall einzelner Komponenten ist für das Gesamtsystem unkritisch, sofern es sich nicht um den **ISOagriNET Parser** oder die **Datenbank** handelt. Fällt eine dieser beiden Komponenten aus, hat dies massive Datenverluste zur Folge. Beide Komponenten arbeiteten jedoch zu jedem Zeitpunkt einwandfrei. Die Gefahr eines auf diese Weise auftretenden Datenverlustes wird daher als gering bewertet.

Die Mehrzahl der im Netzwerk verschickten Nachrichten basiert entsprechend ISOagriNET auf dem verbindungslosen User Datagram Protocol (UDP). Da die Zustellung mittels UDP verschickter Nachrichten nicht garantiert ist, besteht insbesondere bei stark ausgelasteten Netzwerken die Gefahr des Datenverlustes. Die Farming Cell verwendet mittlerweile professionelle **Switches**, da die ursprünglich verwendeten preisgünstigen Geräte den Netzwerkverkehr nur unter Paketverlusten aufrecht erhalten konnten.

Abschließend kann festgehalten werden, dass ausbleibende Datenerfassungen in nahezu allen Fällen auf den Benutzer zurückzuführen sind. Der Betrieb des Gesamtsystems ist aufgrund seiner Vielgliedrigkeit komplex und schwer überschaubar. Die Entwicklung einer als **Überwachungsinstanz** fungierenden Softwarekomponente, welche den Benutzer im Falle ausbleibender Daten benachrichtigt, wäre sinnvoll.

7.1.4 Eignung als wissenschaftliches Werkzeug

Die Eignung der Farming Cell für die Klärung wissenschaftlicher Fragestellungen ist prinzipiell gegeben. Ihre besondere Stärke liegt in der zentralen Datenhaltung und der aus dem Datenmodell resultierenden Möglichkeit, alle erhobenen Informationen miteinander in Beziehung setzen zu können.

Die Erweiterbarkeit des Systems und seine Anpassbarkeit sind im Bereich der Sensorik aufgrund seines modularen Aufbaus gegeben. Durch die individuell festlegbaren Messparameter ist das System an die jeweilige Fragestellung anpassbar.

Der Datenumfang, der von den im Versuchsstall befindlichen Anlagen abgegriffen werden kann, ist dagegen beschränkt. Dies ist darauf zurückzuführen, dass diese nicht für den integrierten Einsatz konzipiert wurden. Im Rahmen des Projektes konnten aus Kapazitätsgründen lediglich Teile des Datenaustausches realisiert werden.
Die Gewinnung tierindividueller Daten ist gegenwärtig, mit Ausnahme der Tierwaage, nicht möglich. Verbesserungspotential besteht somit hinsichtlich der Anlagenanbindung beziehungsweise –auswahl. Die Erweiterbarkeit des Systems um ISOagriNET kompatible Komponenten ist durch dessen Architektur gegeben.

Dem wesentlichen Nutzenaspekt, welcher für die Wissenschaft in der ganzheitlichen Betrachtung des Prozesses besteht, kann somit weitestgehend Rechnung getragen werden.

Bei der Entwicklung aller Systemkomponenten wurde angestrebt, eine einfache Bedien- und Konfigurierbarkeit zu gewährleisten. Aufgrund der strukturell ähnlich gehaltenen Parametrierung der Systemkomponenten, sind die Soft- und Hardwarekomponenten leicht bedienbar.
Der Betrieb des Gesamtsystems hingegen setzt Kenntnisse über Abhängigkeiten der Komponenten und Grundlagenwissen im Umgang mit in Java entwickelter Software und Datenbanken voraus.

7.1.5 Nutzen für landwirtschaftliche Betriebe

Der Nutzen der Farming Cell für Praxisbetriebe ist wesentlich von deren Ausstattung abhängig. Er steigt insbesondere mit der Zahl der vorhandenen Sensoren und Verbrauchsmesser sowie dem Detailgrad tier- oder tiergruppenindividuell generierter Daten. Auch die Anlagenausstattung und deren Interaktionsmöglichkeiten sind maßgeblich.

Ausgehend von einer umfangreichen und integrationsfähigen technischen Ausstattung, entstehen vielfältige Nutzenaspekte für den Landwirt:

- Manuelle und automatisierte Prozessüberwachung (z.B. zeitnahe Benachrichtigung bei kritischen Werten),

- Möglichkeit der Prozessoptimierung (z.B. Variation der Futtermischung auf Basis der Informationen Tageszunahme und Futterverbrauch),
- betriebswirtschaftliche Betrachtungen (z.B. Kostenrechnung),
- Arbeitszeitersparnisse durch automatische Prozessdokumentation sowie durch Prozessunterstützung (z.B. Datenzugriff im Stall),
- automatische Datenweitergabe an Wirtschaftspartner und Behörden sowie Empfang von Nachrichten (z.B. Anmeldung von Schlachttieren und Rückmeldung von Schlachtergebnissen).

Die Farming Cell deckt nicht alle genannten beispielhaften Nutzenaspekte ab. Dies hat zweierlei Gründe. Neben den Möglichkeiten der Einbindung der vorhandenen Technik (vgl. Kapitel 7.1.2), ist die Ausstattung z.B. der Wirtschaftspartner mit Schnittstellen für die Datenentgegennahme und -abgabe ein wichtiger Faktor. Diese Sachverhalte und die mit der Integration verbundenen Kosten sind es, die auch auf einem Praxisbetrieb den tatsächlich möglichen Nutzen beeinflussen.

7.2 Verbesserungsvorschläge

Die verwendeten Standards und die genutzten Hard- wie auch Softwarekomponenten weisen trotz ihres erfolgreichen Einsatzes in der Farming Cell Schwächen auf. Auf diese wird in den folgenden Abschnitten eingegangen.

7.2.1 Standards

Wenngleich der Standard ISOagriNET erfolgreich in der Farming Cell umgesetzt werden konnte, traten Punkte zutage, die verbesserungswürdig sind. Auf diese wird im Folgenden eingegangen. Ebendies gilt für den Standard agroXML.

ISOagriNET

Die Benennung der Entitäten und Items im nationalen (deutschen) Data Dictionary (DD) erfolgt überwiegend in **Deutsch**, Entitäten und Items des internationalen Dictionaries tragen **Englische** Bezeichner. Eine Angleichung wäre sinnvoll.

Eine eklatante Schwäche des **Data Dictionaries** stellt dessen **Struktur** dar. Es ist nicht möglich, eine Entität mit unterschiedlichen Ausprägungen in unterschiedlichen DD Versionen vorzuhalten.

Dies hat zur Folge, dass eine Entität, ist sie einmal veröffentlicht, nicht mehr geändert werden sollte. Im Falle des nationalen Data Dictionaries, welches jahresweise versioniert wird, ist es auf Grund der geschilderten Problematik nicht möglich, Entitäten versionsbezogen zu ändern, hinzuzufügen oder zu entfernen. Änderungen sind immer global und somit für alle Jahrgänge gültig.
Eine weitere Folge ist, dass es bei Verwendung von zwei oder mehr herstellerspezifischen Data Dictionaries zu Kollisionen kommen kann. Jeder Hersteller darf die Nummer für seine Entität aus einem im Standard ISOagriNET definierten Nummernkreis wählen.
Die geschilderte Problematik liegt im **Primärschlüssel** der Entitäten begründet, welcher nur aus der sechsstelligen Nummer (Entity_No) der Entität besteht. Entitäten sollten eine zusätzliche Eigenschaft in Form einer Data Dictionary Zuordnung (zum Beispiel DD_ID) erhalten und der PK in die Primärschlüsselkombination DD_ID und Entity_No geändert werden.
Die geschilderte Problematik gilt ebenso für Items.

Das **Data Dictionary** wird nicht (z.B. durch das LKV NRW) als **Datenbank** frei zugänglich bereitgestellt, sondern ist lediglich in Form von HTML Seiten verfügbar. Hier sollte Abhilfe geschaffen werden, indem die aktuelle Version durch eine noch zu identifizierende Institution im Internet bereitgestellt wird. Auf diese Weise können Inkompatibilitäten vermieden und der entwicklerseitige Aufwand für Implementierungs- und Pflegeprozesse reduziert werden.
Ebenfalls sollte eine öffentlich zugängliche **Plattform** geschaffen werden, die Interessierte über ISOagriNET informiert und **Transparenz** in den Entwicklungsprozess des Data Dictionaries bringt. Gegenwärtig erfolgt dessen Weiterentwicklung und Pflege durch eine in einer Mailingliste organisierte Benutzergruppe. Für Dritte sind aktuelle Entwicklungen nicht einsehbar und Entscheidungsprozesse nicht unmittelbar nachzuvollziehen.

Der Standard ISOagriNET sieht vor, die mittels **Multicast** Mechanismus (vgl. Kapitel 2 und 5.2.1) verschickten Nachrichten nicht mit Headerdaten auszustatten. Da nur im Header die Absenderkennung enthalten ist, kann der Ursprung einer solchen Nachricht empfängerseitig nicht bestimmt werden. Die selbst entwickelten Dienste der Farming Cell senden daher immer Nachrichten mit Header.

In der Praxis ist die Kommunikationsmöglichkeit zwischen zwei ISOagriNET konformen Geräten nicht garantiert, da keine **Mindestinformationsumfänge** definiert sind. Hier muss durch die Anlagenhersteller dringend eine Harmonisierung erreicht werden. Ein mögliches Ziel ist die typweise (Lüftungsanlage, Fütterungsanlage) Festlegung der Entitäten, die entgegengenommen und abgefragt werden können. Existieren keine verbindlichen Festlegungen, kommen die Vorteile der Verwendung eines Kommunikationsstandards nicht zum Tragen.

agroXML

In den Kapiteln 2 und 5.2.2 wurden die durch das KTBL erstellten agroXML Schema Dateien vorgestellt und deren Verwendung im Projekt Farming Cell beispielhaft erläutert. Der dort beschriebene Weg, XSD Dateien in ein Software Framework zu überführen, welches zum Generieren und Verarbeiten von XML Instanzen benutzt werden kann, ist aufwendig. Daher ist es empfehlenswert, derartige Frameworks in verschiedenen Programmiersprachen bereitzustellen. Eventuelle Markteintrittswiderstände und das Entwicklungsfehlerpotential würden so minimiert. Die Entwicklung und die Bereitstellung der **agroXML Frameworks** könnte das KTBL übernehmen.

7.2.2 Hardware

Hohenheimer Messwerterfassung (HME)

Die HME ist ein selbstentwickeltes System, dessen Hardware sich über Monate bewährt hat. Für die Software gilt ebendies mit einer Einschränkung. Die Java Laufzeitumgebung des verwendeten Mikroprozessors besitzt einen, dem Hersteller bekannten jedoch nie behobenen, Fehler im Bereich der **Netzwerkkommunikation**. Auf dem TCP basierende Verbindungen können in Einzelfällen nicht geschlossen werden. Ist die maximal mögliche Anzahl parallel Verbindungen von 25 (Anonymus, 2004b) erreicht, ist der Aufbau einer weiteren Verbindung zur HME unmöglich. Dies betrifft Telnet Verbindungen und das Webinterface. Die Logikkomponente der HME wird hiervon nicht beeinträchtigt, womit die UDP basierende Publikation von Messwerten fortgeführt wird.

Ethernetbox

Die für die Messwerterfassung genutzten Ethernetboxen (vgl. Kapitel 5.4.2) konnten funktional wie auch preislich überzeugen. Ein Manko stellt jedoch deren nicht ISOagriNET konforme **Kommunikation** dar. Um ISOagriNET zu implementieren war daher die Entwicklung eines Gatewaysoftware notwendig (vgl. Kapitel 5.6.3), welche die Kommunikation mit der Ethernetbox durchführt und die Bereitstellung der Daten im Netzwerk übernimmt.

Die Hardwarekomponenten aus denen die Ethernetbox besteht, sind bekannt und die Entwicklung einer ISOagriNET konformen Software für den in der Ethernetbox verbauten **Mikrocontroller** RC2200 der Firma Rabbit Semiconductor[30] möglich. Die sehr gute Hardwarebasis der Ethernetbox könnte in Verbindung mit einer den Standard ISOagriNET implementierenden neuen Mikrocontroller Software als universell nutzbarer ISOagriNET Adapter zum Einsatz kommen.

Wird über die Softwareneuentwicklung hinaus auch eine **Hardwareoptimierung** angestrebt, so sollte ein für den Einsatz im Stall besser geeignetes Gehäuse genutzt werden. Auch wäre es anwenderfreundlich, jeden Kanal als eigene Steckverbindung nach außen zu führen. Die Ethernetbox bietet in ihrer aktuellen Version lediglich vier RJ45 Buchsen für die Belegung der 20 Kanäle.

Datenbankserver

Der Datenbankserver ist ein handelsüblicher Desktop PC (vgl. Kapitel 5.4.3). Es sollte die Anschaffung eines Serversystems in Betracht gezogen werden, welches für die permanente Verfügbarkeit und die Verarbeitung großer **Datenvolumina** ausgelegt ist. Bereits die während eines Mastdurchgangs erfasste Datenmenge führte zu spürbaren Latenzen bei Datenbankoperationen. Die Platzierung eines derartigen Systems auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof würde neben einer verbesserten Performance mehr **Datensicherheit** bieten und könnte auch anderen als Datenablageort dienen. Anpassungen an den Softwarekomponenten der Farming Cell wären im Falle eines Serverwechsel nicht notwendig (MySQL Datenbankserver vorausgesetzt).

Eine Alternative zu der Neuanschaffung eines Datenbankserversystems ist die Verwendung der im Rechenzentrum der Universität Hohenheim verfügbaren

[30] Internetpräsenz: http://www.rabbit.com

Ressourcen (vgl. Kapitel 5.4.3). Hierbei muss jedoch bedacht werden, dass der Transfer der auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof anfallenden Daten in die Datenbank in Hohenheim jederzeit sichergestellt sein muss, anderenfalls droht ein Datenverlust. Da mögliche Abrisse der Internetverbindung auf einer der Seiten oder Vorfälle wie Wartungsarbeiten dieser Bedingung gegenüberstehen, sollte auf der Versuchsstation eine Schicht zur **Datenzwischenspeicherung** implementiert werden. Eine solche Schicht nimmt Daten temporär auf und kann die hierfür verwendeten Ressourcen nach erfolgreicher Übertragung nach Hohenheim wieder freigeben.
Die im Projekt Farming Cell genutzte Datenbank MySQL bietet das Werkzeug der **Replikation** (vgl. Anonymus, 2010a) und somit eine Möglichkeit, Daten automatisiert in eine zweite Datenbank zu übertragen. Da die Datenbank des Rechenzentrums der Universität Hohenheim bereits einem Replikationscluster angehört, könnte diese bei einer derartigen Lösung keine Verwendung finden. Dennoch kann die Infrastruktur des genannten Rechenzentrums genutzt werden. Die bereits vorhandene Datenbank des virtuellen Servers (vgl. Kapitel 5.4.3) ist als Replikationsziel geeignet.

Folgendes **Szenario** ist aus den Überlegungen ableitbar:

- Die Existierende Datenbank auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof wird als Replikationsserver betrieben und dient als Produktivdatenbank.
- Die Datenbank des virtuellen Servers wird Replikationsclient und stellt zum einen eine Sicherung der Produktivdatenbank dar. Zum anderen kann ihr Datenbestand für Lesende Zugriffe, wie beispielsweise Auswertungszwecke, genutzt werden und so die Produktivdatenbank entlasten.

Das beschriebene Szenario erfordert keinerlei Soft- oder Hardwareneuanschaffungen und macht keine Änderungen im Bereich der Datenhaltung erforderlich. Es muss lediglich die Konfiguration der Datenbanken als Replikationscluster vorgenommen werden.

Management PC und Datenbankserver

Die beiden auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof im Einsatz befindlichen Desktop Computer können durch kleinere,

stromsparend arbeitende Systeme ersetzt werden. Die **Substitution** durch ein einzelnes Systems ist denkbar, sofern einer Verlagerung der rechenintensiven Aufgabe des Reportings auf ein System des Rechenzentrums der Universität Hohenheim erfolgt. Ein derartiger Ansatz, ein **Kleinstsystem** zu verwenden, wurde im Rahmen des Projektes verfolgt, führte jedoch aufgrund mangelnder Rechenleistung nicht zum Erfolgt. Das verwendete Kleinstsystem war die NSLU2 der Cisco Tochter Linksys. Inzwischen existieren ähnlich günstige Systeme, die jedoch eine vielfache Rechenleistung bieten. Zu nennen ist in diesem Zusammenhang die Plattform Sheevaplug der Firma Marvell International Ltd., welche ebenso wie das System von Linksys mit einem Debian Betriebssystem ausgestattet werden kann. Somit ist es als Ersatz für Management PC und Datenbankserver nutzbar. Seine Eignung sollte überprüft werden.

Unterbrechungsfreie Stromversorgung (USV)

Um kurzzeitige Stromausfälle und –schwankungen schadfrei überbrücken zu können, sind die auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof befindlichen Geräte Management PC, Datenbank Server und Multigasmonitor durch eine Unterbrechungsfreie Stromversorgung (Modell Back-UPS 650[31] der Firma American Power Conversion Corp.) abgesichert. Diese ist in der Lage, wenige Minuten andauernde **Stromausfälle** zu überbrücken, bietet darüber hinaus jedoch keine Funktionalität. Die Neuanschaffung einer leistungsfähigeren USV mit Funktionen wie E-Mail Benachrichtigung im Falle eines Stromausfalls sowie der Fähigkeit, Rechner zum Herunterfahren zu veranlassen, wäre sinnvoll.

7.2.3 Software

ISOagriNET Parser

Der ISOagriNET Parser ist das wichtigste Daten sammelnde Element der Farming Cell. Er ist in der Lage, mehrere Nachrichten innerhalb einer Minute zu zerlegen und die gewonnenen Informationen in der Datenbank zu speichern. Wird eine gegenwärtig unbekannte Anzahl Nachrichten pro Zeiteinheit überschritten, wird der Parser nicht mehr in der Lage sein, die Nachrichten zu verarbeiten und

[31] Produktinformationen:
http://www.apc.com/resource/include/techspec_index.cfm?base_sku=bk650mc

Datenverluste sind die Folge. Dies kann bereits bei einer kurzzeitigen Erhöhung der Nachrichtenanzahl eintreten.

Der beschriebenen Gefahr kann begegnet werden, indem die Funktionen (Nachrichtenempfang, -zerlegung, Datenbankschreibzugriff) des Parsers in mindestens zwei Gruppen aufgeteilt werden und eine **quasi-parallele Abarbeitung** mit einer dazwischen geschalteten Warteschlange erfolgt. Zu beachten ist, dass die Aufgabengruppe des Nachrichtenempfanges mit anschließendem Ablegen in der Warteschlange oberste Priorität genießt. Die zweite Aufgabengruppe (Nachrichtenzerlegung und Datenbankschreibzugriff) wird nur dann ausgeführt, wenn die erste Gruppe pausiert, d.h. keine Nachrichten empfangen werden.

Zur besseren Überprüfbarkeit des Warteschlangenfüllstandes wäre ferner die Ausgabe des Füllstandes in einer Grafischen Oberfläche (engl. Graphical User Interface, GUI) oder in der Eingabeaufforderung sinnvoll.

Die genannten Verbesserungsvorschläge sollten, insbesondere im Falle der Umsetzung des vorgeschlagenen **Datenbank Redesigns**, beherzigt werden, da dieses den Verarbeitungsaufwand auf Seiten des ISOagriNET Parsers erhöhen wird.

Datenbankstruktur

Einen zentralen Bereich des Datenbankschemas bilden die Messdaten. Auf das Design des in Abbildung 7.1 dargestellten Teilbereiches soll an dieser Stelle eingegangen werden.

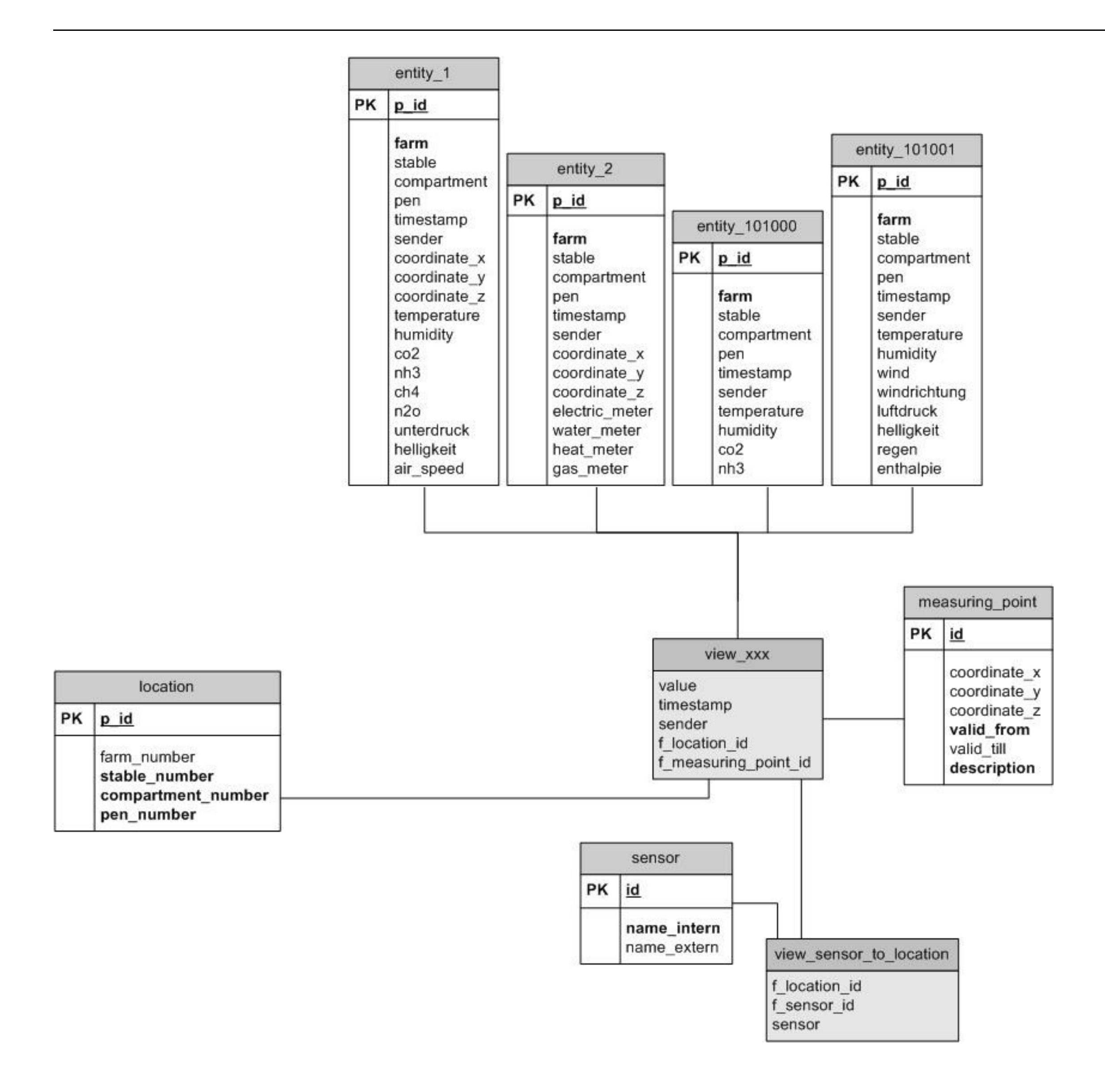


Abbildung 7.1: Datenbank Schema – Auszug aus Cluster 1 - Mess- und Anlagendaten

Die hier dargestellte **Struktur** orientiert sich an der des ISOagriNET Data Dictionaries, was an den Tabellen entity_1, entity_2, entity_1010100 und entity_101001 deutlich wird. Die Entscheidung, das ADED in der Datenbank nachzubilden, liegt darin begründet, dass die zahlreich beim ISOagriNET Parser eingehenden Nachrichten auf diese Weise ohne weitere Bearbeitung in die Datenbank geschrieben werden können. Das Überführen in eine zweite Datenstruktur entfällt. Dieser Vorteil birgt jedoch den Nachteil, dass gleichartige Messwerttypen, wie beispielsweise die in den Tabellen entity_1, entity_2 und entity_101000 vorkommenden Temperaturwerte, zu Auswertungszwecken im Nachhinein zusammengeführt werden müssen. Dieses ist für den Benutzer insofern unkritisch, als dass entsprechende **Views** ihm diese Aufgabe abnehmen (vgl.

Abbildung 7.1). Der Zeitaufwand für das Erstellen dieser dynamisch generierten Views ist jedoch durch die Vielzahl Datensätze enorm und bei jedem Zugriff erneut durchzuführen. Entsprechend langsam ist das Arbeiten mit den Views.
Um der Problematik zu begegnen, können zwei **Optimierungsansätze** verfolgt werden, welche beide auf die Verwendung von Views verzichten.

1. Optimierung des Datenbankdesigns für Auswertungszwecke, sowie Anpassung des vorgelagerten ISOagriNET Parsers und aller Datenbankschnittstellen der nachgelagerten Dienste.
2. Ersetzen der Views durch strukturell identische Tabellen.

Variante eins verfolgt mit einem Redesign der Datenbank einen konsequenten, wenngleich tiefgreifenden Ansatz. Umfangreiche Änderungen in der mittleren und oberen Software Layer (vgl. Abbildung 5.1) wären vonnöten. Würden die Datenbank und der ISOagriNET Parser in der Form angepasst, dass die empfangenen ADIS/ADED Nachrichten zerlegt und deren enthaltene Werte separiert nach Typ (Temperatur, Wassermesser etc.) in getrennte Tabellen gelegt würden, erhielte man eine Vielzahl an Tabellen, vergleichbar mit den derzeit existierenden Views. Der bereits erläuterte Vorteil käme bei allen späteren Zugriffen auf die Daten zum tragen. Bedacht werden muss aber, dass der ISOagriNET Parser gegenüber der aktuellen Implementierung ein Mehr an Arbeit zu verrichten hätte. Es muss sichergestellt sein, dass das Zerlegen und die Mehrzahl an Schreiboperationen durch den ISOagriNET Parser geleistet werden können. Andernfalls bestünde die Gefahr eines Datenverlustes (vgl. auch nachfolgende Betrachtung des ISOagriNET Parsers). Ferner ist zu beachten, dass die Datenbankschnittstellen vieler nachgelagerter Dienste der oberen Software Layer geringfügig angepasst werden müssten.

Variante zwei sieht vor, das Datenbankdesign nahezu beizubehalten. Lediglich die Views wären durch identisch aufgebaute Tabellen zu ersetzen. Zwar würde sich, durch die Redundanz der Daten, das Volumen der Datenbank nahezu verdoppeln, der Vorteil, performantere Auswertungsmöglichkeiten zu erhalten und die Tatsache, dass Speicherplatz nicht als kritischer Faktor anzusehen ist, überwiegen jedoch. Das Befüllen der Tabellen, welche an die Stelle der derzeit vorhandenen Views treten würden, stellt bei dieser Variante die zentrale Herausforderung dar. Die Eigenschaft

der Views, jederzeit alle in der Datenbank befindlichen Messwerte darzustellen, müsste nachgebildet werden. MySQL bietet mit Stored Programs and Views (Anonymus, 2010b) Möglichkeiten, die Tabellen regelmäßig datenbankseitig automatisch fortschreiben zu lassen.
Die vor- sowie nachgelagerte Software könnte bei Variante zwei unverändert bleiben, sofern Struktur und Nomenklatur der vorhandenen Views für die neu anzulegenden Tabellen übernommen würden.

Aufgrund der umfangreicheren Anpassungen die Variante eins erfordert, wird im Fall der Farming Cell die Umsetzung von Variante zwei empfohlen. Wird die Farming Cell auf andere Betriebe oder Ställe übertragen, sollte Variante eins von Beginn an implementiert werden.
Wird die derzeitige Variante, d.h. das Arbeiten mit Views, beibehalten, kann durch die Nutzung performanterer Serverhardware eine Verbesserung hinsichtlich des Zeitaufwandes für die Erstellung der Views erreicht werden. Auch das Herabsetzen der Messwerterfassungsintervalle und somit des Datenvolumens beschleunigt das Operieren mit den Views.

Webapplikation

Die Benutzerschnittstelle für das Prozessmanagement, die Webapplikation, ermöglicht es, Daten potentieller Masttiere zu importieren (vgl. Kapitel 5.6.8). Die zu übernehmenden Daten müssen im **Excel** Format vorliegen, da die Managementsoftware Supersau, die in dem der Mast vorgelagerten Prozess Anwendung findet, dieses als Exportformat vorsieht. Da wiederholt Schwierigkeiten während des **Importvorganges** auftraten, welche auf das Excel Format zurückzuführen sind, sollte die Importschnittstelle überarbeitet werden. Bereits kleine, teilweise nicht in Excel zu beeinflussende Änderungen am Format, führen zu Fehlern beim Import. Es wird daher empfohlen, die Importschnittstelle auf das **CSV** Format umzustellen. Auf diese Weise kann die Exportfunktion der Supersau weiterhin genutzt werden, indem die exportierten Excel Dateien beispielsweise mit Microsoft Excel in das CSV Format überführt und anschließend importiert werden.

Reporting Applikation

Die zum Zwecke der Darstellung gesammelter Messwerte implementierte Anwendung konnte nur bedingt überzeugen.

Das verwendete Framework BIRT ermöglicht es, Abbildungen zu generieren (vgl. Kapitel 5.6.9.1), verbraucht jedoch viele Ressourcen. In Einzelfällen ist die Rechenleistung des Webservers nicht ausreichend. Das zuvor dargestellte Redesign der Datenbank verspricht diesbezüglich eine Verbesserung.

Von der weiteren Verwendung des Frameworks BIRT wird dennoch abgeraten. Stattdessen sollten schlankere Lösungen, wie beispielsweise die für den Info Mailer verwendete Google Chart API (vgl. Kapitel 5.6.7), in Betracht gezogen werden. Ebenso wäre es denkbar, Auswertungsaufgaben automatisiert in der Nacht anzustoßen und die dann freien Ressourcen des Rechenzentrums der Universität Hohenheim in Anspruch zu nehmen.

7.3 Bewertung des gewählten Vorgehens

Zu Beginn des Projektes erfolgten Festlegungen hinsichtlich der Entwicklungs- und Implementierungsmethode. Auch erfolgte eine Technologieauswahl. Die folgenden Seiten reflektieren die Eignung der gewählten Methoden und Werkzeuge.

7.3.1 Entwicklungs- und Implementierungsmethode

Das Kapitel 4.1 hat sich mit der Auswahl und der Vorstellung der **Entwicklungsmethode** befasst. Die gewählte Methode Crystal Clear zeichnet sich vor allem durch Transparenz sowie formlose und stetige Kommunikation zwischen Kunde und Entwickler aus. Sie konnte erfolgreich gelebt werden und bildete die Grundlage für den Projekterfolg. Insbesondere der stetige Ideenaustausch zwischen den Projektbeteiligten und die formlose Arbeitsweise beschleunigten das Fortschreiten.

Die Kommunikation mit Dritten, gemeint sind beispielsweise Firmen und kooperierende Projektteams, verlief, das Paradigma der Offenheit verfolgend, überwiegend gut. Die Methode Crystal Clear ist auch hier in der Lage, das Projekt zügig voranzubringen, jedoch setzt sie viel Vertrauen zwischen den Beteiligten voraus. Durch die zumeist informelle Kommunikation und fehlende Dokumentation von zum Beispiel Aufgabenpaketen, fehlt ein Kontrollinstrument. Eine formal stärker geprägte Methode könnte diesbezüglich Vorteile bieten.

Abschließend kann festgehalten werden, dass die Entwicklungsmethode Crystal Clear innerhalb des Projektteams Farming Cell erfolgsfördernd war. Ihre Eignung für projektübergreifende Entwicklungen ist unter anderem anhand des Kriteriums Vertrauen individuell zu prüfen.

Neben der Entwicklungsmethode nennt Kapitel 4.1 mit Experimentellem Prototyping auch die gewählte **Implementierungsmethode**. Sie wurde gewählt, da sie schnell vorliegende Prototypen verspricht und sich gut mit dem experimentellen Charakter der Farming Cell vereinbaren lässt. Die Methode war im Falle der Farming Cell zielführend, da der Entwicklungszeitraum knapp bemessen war. Die schnell erfolgten Implementierungen lauffähiger Prototypen durchliefen nur kurze Testphasen, bevor sie eingesetzt wurden. Eine höhere Fehleranfälligkeit der Anwendungen musste hingenommen werden. Nachbesserungsarbeiten waren die Folge, nahmen aber wenig Zeit in Anspruch, da sie im Rahmen der iterativ durchgeführten Erweiterung der Prototypen durchgeführt werden konnten.
Die Wahl einer alternativen Implementierungsmethode, die umfassende Planungsphasen vor der eigentlichen Implementierung vorsieht, wäre für die Farming Cell wenig geeignet gewesen.

7.3.2 Technologie

In diesem Kapitel wird auf die Eignung der zwei zentralen Elemente Programmiersprache und Datenbanksystem eingegangen. Die Kapitel 4.2 und 5.6.6 nennen die Kriterien, die zu deren Wahl führten.

Als **Programmiersprache** kam Java zum Einsatz. Sowohl Konsolenanwendungen und die Webapplikation, als auch der Mikrocontroller der HME sind in Java programmiert. Die Wiederverwendbarkeit von Quellcode beschleunigte den Entwicklungsprozess und die Plattformunabhängig von Java Software ermöglichte den Betrieb auf den unterschiedlichen Systemen der Farming Cell. Mit seinem Funktionsumfang genügte Java den Anforderungen.
In Einzelfällen ist die Verwendung anderer Programmiersprachen von Vorteil, da beispielsweise in C implementierte Bibliotheken durch Hersteller für die Anlagenkommunikation bereitgestellt werden. Ein Beispiel für einen solchen Fall ist die Kommunikation zwischen Computer und Fütterungssteuerung. Diese erfolgt mithilfe einer in C implementierten Bibliothek des Herstellers, die aus einer Java

Anwendung heraus genutzt wird (vgl. Kapitel 5.6.1). Konstellationen wie diese erfordern bei Implementierungen in Java einen Mehraufwand, dieser ist jedoch überschaubar.

Das gewählte **Datenbanksystem** (DBS), bestehend aus MySQL und phpMyAdmin, erfüllte seine Aufgaben den Erwartungen entsprechend. Es sollte, insbesondere aufgrund der Tatsache dass das Rechenzentrum der Universität Hohenheim ein identisches zur Verfügung stellt, beibehalten werden (vgl. Kapitel 5.6.6). Die Notwendigkeit für den Austausch des DBS gegen ein anderes, unter Umständen nicht kostenloses, wird gegenwärtig nicht gesehen. Ebenso wie MySQL sind jedoch auch andere frei verfügbare Datenbanken geeignet, den Anforderungen der Farming Cell Rechnung zu tragen.

7.4 Ausblick

Das Potential, welches ein ISOagriNET Netzwerk wie die Farming Cell durch die Informationszusammenführung und die daraus resultierenden Möglichkeiten der Prozessüberwachung, -unterstützung und -auswertung bietet, ist groß. Als Prototyp zeigt die Farming Cell viele Möglichkeiten auf, wenngleich sowohl hinsichtlich der Architektur der Farming Cell, als auch bezüglich der verwendeten Standards und deren Etablierung Optimierungspotentiale identifiziert werden konnten.
Insbesondere ist eine Einigung der Anlagenhersteller auf verbindlich zu implementierende Befehlssätze bisher nicht erfolgt, obwohl dies als aktuell wichtigster Schritt anzusehen ist. Erst wenn diesbezüglich Einigkeit herrscht, ist die Hersteller übergreifende Interoperabilität gegeben und der Standard ISOagriNET kann sein Potential entfalten. Gegenwärtig sind Neuentwicklungen geprägt von bilateralen Lösungen, die zumeist ISOagriNET konform sind, jedoch individuelle Anforderungen und Anlageneigenschaften einzelner Hersteller berücksichtigen. Dies widerspricht der Intention, allgemeingültige Schnittstellen zu schaffen.

Die Bemühungen, Hard- und Software landwirtschaftlicher Betriebe zu vernetzen, reicht langfristig nicht aus. Der angestrebte Einsatz von Standards sollte als Chance genutzt werden, innovative Bewegungen der IKT Branche aufzugreifen. Zu nennen sind hier zum einen Cloud Computing, welches viele Vorteile für Systementwickler und -nutzer birgt und als junges Konzept bewährte Technologien neu vereinigt. Zum

anderen rücken Versorgungsunternehmen für Gas, Strom und Wasser die Notwendigkeit der Verbrauchsüberwachung (Smart Metering) und den sich daraus ergebenden Nutzen in den Vordergrund. Die Landwirtschaft mit ihrem umfangreichen Ressourceneinsatz sollte auch dieses Thema aufgreifen.

Der Landwirtschaft bietet sich durch die einsetzende Innovationsbereitschaft in der Branche gegenwärtig die Möglichkeit, ein Technologietreiber zu werden. Voraussetzung hierfür ist jedoch, dass repräsentative Projekte die obigen Gedanken und den Ansatz der Farming Cell aufgreifen. Andernfalls besteht die Gefahr, dass die Bereitschaft zu grundsätzlichen Veränderungen versandet und die Nutztierhaltung weiterhin ein technologisches Schattendasein führt.

8 Zusammenfassung

Mit der Farming Cell ist im Versuchsstall für Mastschweine auf der Versuchsstation für Tierhaltung, Tierzüchtung und Kleintierzucht / Unterer Lindenhof der Universität Hohenheim ein Prototyp entstanden, der die Möglichkeiten des Technologieeinsatzes zum Zwecke der Prozessdokumentation und –unterstützung in der Schweinemast aufzeigt. Ferner lieferten der Entwicklungsprozess und der Testbetrieb wichtige Erkenntnisse über Herausforderungen, die es bei einer derartigen Umsetzung zu meistern gilt.
Aufgrund ihrer Einzigartigkeit kann die Farming Cell als Referenz für zukünftige Agrarinformationssysteme dienen. Ihre Stärke ist neben der konsequenten Umsetzung von ISOagriNET ihr modularer Aufbau. Dieser ermöglicht es, einzelne Komponenten des Systems auszutauschen oder auf andere Anwendungsfälle zu übertragen.

Die aus der Heterogenität in der Anlagenlandschaft resultierende Vielzahl oftmals proprietärer Hard- und Softwareschnittstellen verdeutlicht die Notwendigkeit der Standardisierung, welcher mit dem Standard ISOagriNET Rechnung getragen wird. Seine Implementierung lieferte die Erkenntnis, dass seine Eignung gegeben ist, in Teilaspekten jedoch Optimierungspotential besteht. Dies gilt zum einen für die Inhalte des Standards und zum anderen für dessen Erweiterungsprozess.
Die Hersteller technischer Komponenten bestimmen maßgeblich die Geschwindigkeit der Marktdurchdringung mit ISOagriNET kompatiblen Anlagen, implementieren den Standard derzeit jedoch noch zögerlich. Die Alternative der Umsetzung nicht Standard konformer Schnittstellen auf ISOagriNET durch Dritte, wie es in der Farming Cell erfolgte, stellt in der Praxis keine Alternative dar.
Besser sind die Möglichkeiten, Umweltparameter und Ressourcenverbräuche ISOagriNET konform zugänglich zu machen. Das Nach- oder Umrüsten von Sensoren und Verbrauchsmessern ist kostengünstig möglich. Angeschlossen an ISOagriNET implementierende Gateways wie sie im Projektverlauf entstanden, können ihre Messwerte im Netzwerk bereitgestellt werden.

Der für den außenwirtschaftlichen Datenaustausch eingesetzte Standard agroXML befindet sich noch am Anfang seiner Entwicklung. Bis dato sind lediglich Teilbereiche

des tierischen Produktionsprozesses durch das KTBL modelliert worden. Die bisherigen Ansätze sind vielversprechend, die Problematik der Marktdurchdringung besteht jedoch auch hier. Ebenso wie für ISOagriNET sind auch für agroXML praktische Umsetzungen kurzfristig anzustreben.

9 Summary

The Farming Cell is a prototype, showing the prospects of information technology for process documentation and process support. The development and the long time tests in a pig fattening barn at the Research Station for Animal Husbandry, Animal Breeding and Small Animal Breeding / Unterer Lindenhof of the Universität Hohenheim demonstrated the challenges about setting up a farm network. Because of its uniqueness, the Farming Cell can serve as a reference implementation for future agricultural information systems. Its strengths are the consistent utilization of ISOagriNET and a modular design, which offers possibilities to exchange individual components or to transfer them to another farm.

Many challenges occurred during implementation. There are heterogeneous systems with a variety of proprietary hardware and software interfaces. The standard ISOagriNET focuses on integration of different devices using a standardized communication. During its implementation within the Farming Cell, the suitability of ISOagriNET was verified and potentials for optimization have been discovered. With this standard it is easy to integrate sensors and meters into a system. Connected to ISOagriNET gateways like those developed, their data can be provided in the network. Manufacturer of technical components like feedings and climate controls hesitate to implement ISOagriNET for many reasons. Therefore they limit the speed of its market penetration.

For external data exchange the standard agroXML was used. It is still under development and only small parts of the animal production process have been modeled. The current approaches are promising, but market penetration is also challenging. Exemplary implementations are essential for the future success of ISOagriNET and agroXML.

10 Literaturverzeichnis

Anonymus (1998): MEGACOMP GENPRO REIN-RAUS TURBOCLEAN. Schauer Maschinenfabrik GmbH & Co KG.

Anonymus (2004a): XR3000 Benutzerhandbuch. Tru-Test Limited.

Anonymus (2004b): Getting started with TINI. Maxim Integrated Products.

Anonymus (2006): ASA Antenna panels for harsh Environment. Agrident GmbH.

Anonymus (2008): ASR700/766 – The long-range reader with DSP. Agrident GmbH.

Anonymus (2007): Felddaten - Schweinemast (Wirtschaftsjahr 2006/2007). ZDS - Zentralverband der Deutschen Schweineproduktion e.V., http://www.erzeugerring.info/index.php?con_cat=5&con_lang=1&sid=a045229b ecea84d0a6e9d58b0592a614&mr=1&sch=14, 20.11.2009.

Anonymus (2009a): ISO 17532: Stationary equipment for agriculture - Data communications network for livestock farming. Genf: Beuth Verlag.

Anonymus (2009b): ISOagriNET-LON-Adapter. Möller GmbH. Diepholz.

Anonymus (2009c): The MultiSensor Users Guide, Sensors for the 1-Wire Bus, All current models of MultiSensor. iButtonLink LLC, Dallas.

Anonymus (2010a): Replikation bei MySQL. Oracle Corporation, http://dev.mysql.com/doc/refman/5.1/de/replication.html, 26.02.2010.

Anonymus (2010b): Stored Programs and Views. Oracle Corporation, http://dev.mysql.com/doc/refman/5.0/en/stored-programs-views.html, 03.03.2010.

ADRION, F. (2009): Untersuchungen zur Eignung und Messgenauigkeit eines Multigasmonitors. Bachelorarbeit, Universität Hohenheim, Institut für Agrartechnik, unveröffentlicht.

ANCUTICI, M. (2009): Datenbank. Universität Hohenheim, https://rz.uni-hohenheim.de/datenbank.html, 27.11.2009.

BIRGELS, F. (2009): Virtuelle Server. Universität Hohenheim, https://rz.uni-hohenheim.de/vserver.html, 17.02.2009.

BEA, W. (2004): Vergleich zweier Mastschweinehaltungssysteme - Beurteilung der Tiergerechtheit. Dissertation Universität Hohenheim. Forschungsbericht Agrartechnik der VDI-MEG Nr. 419.

BECK, K., BEEDLE, M., VAN BENNEKUM, A., COCKBURN, A., CUNNINGHAM, W., FOWLER, M., GRENNING, J., HIGHSMITH, J., HUNT, A., JEFFRIES, R., KERN, J., MARICK, B., MARTIN, R. C., MELLOR, S., SCHWABER, K.,

SUTHERLAND, J., THOMAS, D. (2001): Manifesto for Agile Software Development. Ward Cunningham, http://agilemanifesto.org, 13.01.2010.

BERNHARD, H., HECKMANN, M. (2008): Dokumentationsaufwand in der Mastschweineproduktion. Landtechnik 2008, H. 3, S. 164-165.

BLE (2008): Versorgungsbilanz Fleisch 2003 bis 2005 und Regionale Versorgungsbilanz Fleisch 2006. BLE, http://www.ble.de/cln_099/nn_417426/SharedDocs/Downloads/06__Aktuelles/06__Publikationen/Broschueren/VersorgungsbilanzFleisch,templateId=raw,property=publicationFile.pdf/VersorgungsbilanzFleisch.pdf, 24.11.2009.

BÜSCHER, W., RUDOVSKY, A., MARKS, M., HAEUSER, S., HESSE, D. (2008): DLG-Merkblatt 351, Tränketechnik für Schweine. Frankfurt am Main.

• COCKBURN, A. (2005): Crystal Clear, Agile Software-Entwicklung für kleine Teams. Mitp-Verlag: Frechen.

• DERN, G. (2006): Management von IT-Architekturen - Leitlinien für die Ausrichtung, Planung und Gestaltung von Informationssystemen. Vieweg: Wiesbaden.

DLG (2009): DLG und BFL haben erstmals Zertifikate ISOagriNET conform verliehen. Proplanta GmbH & Co. KG, http://www.agrar-presseportal.de/Nachrichten/agrar_presseportal_nachricht_pdf_zip.php?id_S=3282, 26.10.2009.

DOLUSCHITZ, R., KUNISCH, M. JUNGBLUTH, T., EIDER, C. (2005): agroXML – A standardized Data Format for Information Flow in Agriculture. EFITA/WCCA2005 Joint Congress on IT in Agriculture, Vila Real, Portugal, 25.-28.7.2005, European Federation for Information Technology in Agriculture, Food and Environment, S. 439 - 443.

GALLMANN, E. (2003): Vergleich von zwei Haltungssystemen für Mastschweine mit unterschiedlichen Lüftungsprinzipien – Stallklima und Emissionen. Dissertation Universität Hohenheim. Forschungsbericht Agrartechnik der VDI-MEG Nr. 404.

• GÖTZENAUER, J. (2006): Agile Methoden in der Softwareentwicklung: Vergleich und Evaluierung. Grin Verlag: München.

GROENEVELD, E. (2002): An adaptable platform independent information system in animal production: framework and generic database structure. Livestock Production Science 2002, No. 87, pp. 1-12.

GÜTTICH, G. (2004): Im Test: Mess-PC von Better Networks, Modulares Überwachungssystem. IT Administrator, Das Magazin für professionelle System und Netzwerkadministration 2004, H. 11, S. 19-21.

HÄUSSERMANN, A. (2006): Stallklimaregelung und Emissionen – Entwicklung und Evaluierung sensorgestützter komplexer Regelstrategien für die Mastschweinehaltung. Dissertation Universität Hohenheim. Forschungsbericht Agrartechnik der VDI-MEG Nr. 443.

• HEINRITZI, K., GINDELE, H. R., REINER, G., SCHNURRBUSCH, U. (2006): Schweinekrankheiten. Utb: Stuttgart.

HENNINGER, G. (2007): ISOBUS - die Zukunft hat längst begonnen. Landtechnik 2007, H. 4, S. 206-207.

HERD, D., GALLMANN, E., RÖSSLER, B., JUNGBLUTH, T. (2007): Rückverfolgbarkeit in der Schweinehaltung. Landtechnik 2007, H. 6, S. 410-411.

HERD, D., KUHLMANN, A., MARTINI, D., KUNISCH, M., FRIEDRICHS, E. (2008): Technische Möglichkeiten zur Verbesserung der Prozessdokumentation und Rückverfolgbarkeit in der Schweinehaltung. Precision Pig Farming: Innovative Technologien und Entscheidungsmodelle für die Schweinehaltung, Osnabrück, Deutschland, 30.9.-1.10.2008, KTBL, S. 121-131.

HOLPP, M. (2008): Software in der Tierhaltung, in Vorlesungsunterlagen Software in der Landwirtschaft der Forschungsanstalt Agroscope Reckenholz-Tänikon, Ettenhausen.

• HORSTMANN, J. (2006): Open Source Jahrbuch 2006: Freie Datenbanken im Unternehmenseinsatz: Analyse und Vergleich der wichtigsten Open-Source-Datenbanken. Lehmanns Media: Berlin.

• KERSKEN, S. (2009): IT-Handbuch für Fachinformatiker. Galileo Press: Bonn.

• KTBL (2006): Nationaler Bewertungsrahmen Tierhaltungsverfahren. KTBL: Darmstadt.

KUHLMANN, A., HERD, D., RÖSSLER, B., GALLMANN, E., JUNGBLUTH, T. (2009): Development and evaluation of two ISOagriNET compliant systems for measuring environment and consumption data in animal housing systems. Joint International Agricultural Conference 2009, Wageningen, Niederlande, 6.-8.7.2009, S. 99-106, EFITA.

KUHLMANN, A., RÖSSLER, B. (2009): Hohenheimer Messwerterfassung – HME, Benutzerhandbuch. Universität Hohenheim.

KUNISCH, M., FRISCH, M., MARTINI, D., BÖTTINGER, S. (2007): agroXML – a standardized language for data exchange in agriculture. EFITA / WCCA 2007, Environmental and rural sustainability through ICT, Glasgow, Schottland, 2.-5.7. 2007, EFITA.

MARTINI, D. (2007): Gemeinsame Datenstruktur ISOagriNet und agroXML. IT Food Trace Meilensteinbericht des KTBL vom 15. August 2007, Darmstadt.

MARTINI, D., SCHMITZ, M, Kunisch, M. (2008): Erweiterung von agroXML zur Dokumentation und Qualitätssicherung in der Tierhaltung. eZAI 2008, H. 8.

NOACK, P. O. (2007): Standards für den elektronischen Datenaustausch in der Landwirtschaft. Landtechnik 2007, Sonderheft, S. 283-285.

NOURIE, D., PAWALAN, M. (2007): New to Java Programming - Introducing the Java Platform. Oracle Corporation, http://java.sun.com/new2java/programming/intro, 14.01.2010.

PAULSEN, C., SPILKE, J., WAGNER, P. (2005): ISOagriNet – ein ISO-Projekt zur Standardisierung der elektronischen Kommunikation in der Erzeugung tierischer Produkte und deren Verarbeitung. eZAI 2005, H. 2, S. 25-26.

• POMBERGER, G., PREE, W. (2004): Software Engineering: Architektur-Design und Prozessorientierung. Hanser Fachbuchverlag: München.

SCHULZE, L. (2010) : Ethernetbox Netzwerkprotokoll. better networks, http://www.messpc.de/netzwerkprotokoll.php, 24.11.2009.

• SEEBOERGER-WEICHSELBAUM, M. (2004): Java Server Pages Kompendium. Markt und Technik Verlag: München.

SEIDLER, K. (2009): XAMPP. Apache Friends, http://www.apachefriends.org/de/xampp.html, 14.01.2010.

SPILKE, J., ZÜRNSTEIN, K. (2005): Webservices - Beschreibung eines Ansatzes zur Anwendungskopplung und von Nutzungsmöglichkeiten im Agrarbereich. eZAI 2005, H. 2, S. 33-40.

STEINBERGER, G., ROTHMUND, M., MARTINI, D., SPIETZ, C., MALLON, D., NASH, E. (2007): Integration von agroXML in eine landwirtschaftliche Geodateninfrastruktur. Landtechnik 2007, Heft 2, S. 114-115.

STEINBERGER, G., ROTHMUND, M. AUERNHAMMER, H. (2009): Mobile farm equipment as a data source in an agricultural service architecture. Computers and Electronics in Agriculture 2009, No. 65, pp. 238-246.

VON BORELL, E, BOKISCH, F.-J., BÜSCHER, W., HOY, S., KRIETER, J., MÜLLER, C., PARVIZI, N., RICHTER, T., RUDOVSKY, A., SUNDRUM, A., VAN DEN WEGHE, H. (2001): Critical control points for on-farm assessment of pig housing. Livestock Production Science 2001, No. 72, pp. 177-184.

• WALLMÜLLER, E. (2001): Software-Qualitätsmanagement in der Praxis: Software-Qualität durch Führung und Verbesserung von Software-Prozessen. Carl Hanser Fachbuch: München.

ZALUDIK, K. (2002): Bewertung praxisüblicher Mastschweinehaltungen in Nordrhein-Westfalen hinsichtlich der Tiergerechtheit. Dissertation Universität Hohenheim. http://opus.ub.uni-hohenheim.de/volltexte/2002/20/.

Anhang

A – Ressourcen und Systemarchitektur

Gesamtsystem

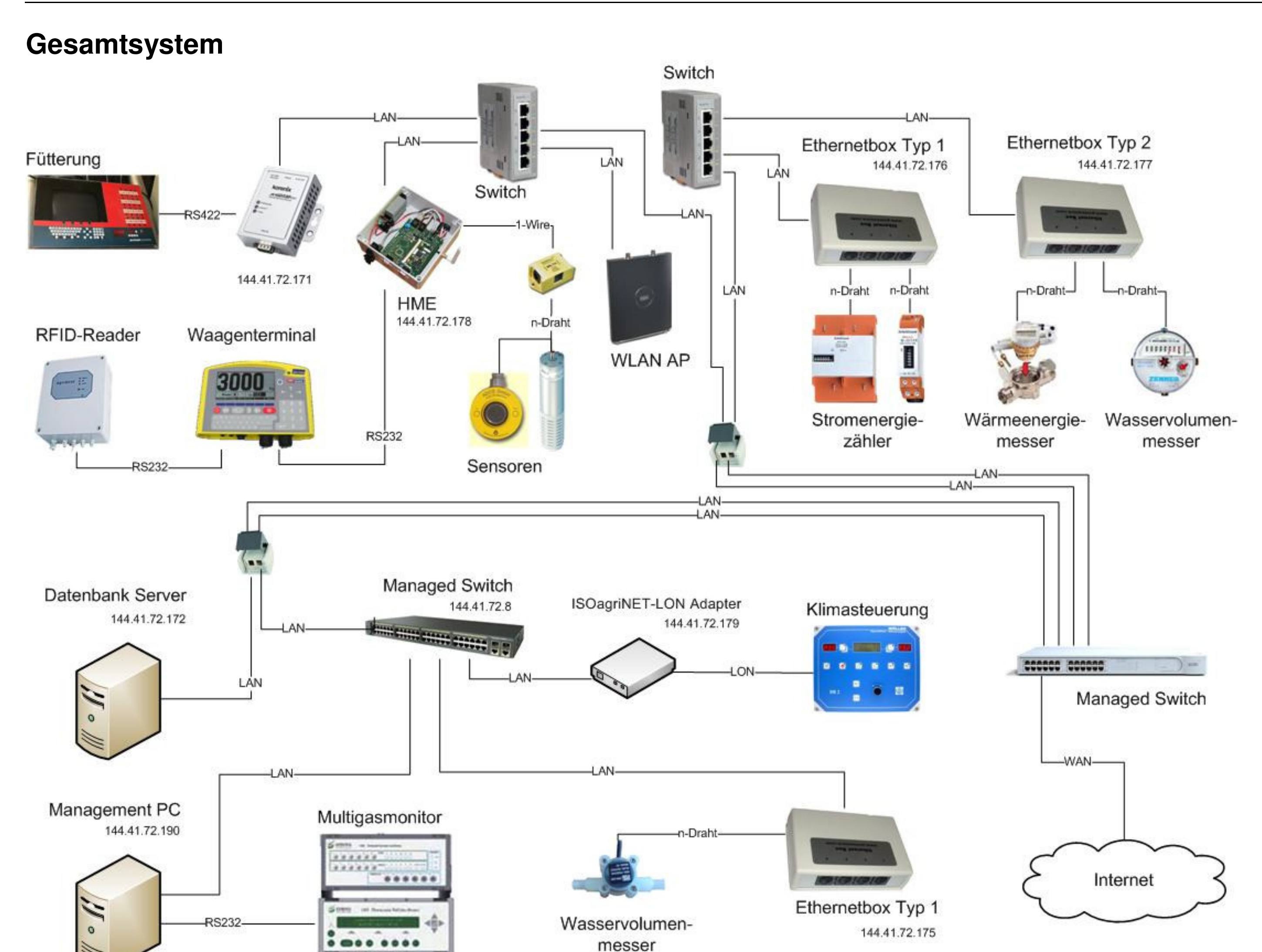


Abbildung 0.1: Hardwareüberblick

Die Abbildung zeigt schematisch die Hardwarearchitektur der Farming Cell in ihrer größten Ausbaustufe.

Tabellendefinitionen

Auf den folgenden Seiten sind die Strukturen aller Tabellen der Farming Cell Datenbank in alphabetischer Reihenfolge hinterlegt. Dies umfasst die Nennung aller Felder (auch Attribute genannt), deren Typ sowie deren Besonderheiten und einen kurzen beschreibenden Text. Tiefergehende Erläuterungen zu jedem einzelnen Attribut folgen in einem separaten Abschnitt.

Abkürzungen

PK = Primary Key; Primärschlüssel oder die Primärschlüsselkombination einer Tabelle
FK = Foreign Key; Fremdschlüssel aus einer fremden Tabelle
DV = Default Value; Verwendeter Standardwert wenn kein anderer gesetzt wurde

animal

Alle aktuellen und ehemaligen Masttiere.

Feld	Typ	Null	Besonderheit	Kommentar
p_electronical_id	bigint(15)	Nein	PK	Nummer der RFID Ohrmarke
visual_id	int(11)	Nein		Visuelle ID der Ohrmarke
f_father_id	int(11)	Ja	FK	ID des Ebers aus Tabelle boar
mother_no	varchar(10)	Nein		Nummer der Mutter
piglet_no	tinyint(4)	Nein		Nummer des Ferkels
date_of_birth	date	Ja		Geburtsdatum
sex	varchar(2)	Ja		Geschlecht; 1 = männlich 2 = weiblich
f_race_id	tinyint(4)	Ja	FK	ID aus Tabelle race
waiting_time	date	Ja		Datum bis zu dem Tier gesperrt ist
arrived_timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Zeitpunkt Einstallen / Mastbeginn
left_timestamp	timestamp	Ja		Zeitpunkt Ausstallen
f_leaving_reason_id	tinyint(4)	Ja	FK	ID des Abgangsgrundes
f_leaving_contact_id	int(11)	Ja	FK	ID des Kontaktes der abgehendes Schwein aufnimmt

animal_to_location

Zuordnung aller Masttiere zu Lokationen.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	bigint(20)	Nein	PK	Fortlaufende Nummer
f_e_animal_id	varchar(16)	Nein	FK	Nummer der RFID Ohrmarke aus Tabelle animal
f_location_id	tinyint(4)	Nein	FK	ID der Lokation aus Tabelle location
arrived_timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Zeitpunkt an dem das Tier die Lokation betreten hat
left_timestamp	timestamp	Ja	NULL	Zeitpunkt an dem die Lokation verlassen wurde
f_user_id	tinyint(4)	Nein	FK	ID des durchführenden Benutzers

boar

Alle bekannten Eber.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	int(11)	Nein	PK	Fortlaufende Nummer
name	varchar(64)	Nein		Name des Ebers(aus Supersau Datei)
mother_no	varchar(10)	Ja		Nummer der Muttersau
piglet_no	tinyint(4)	Ja		Fortlaufende Ferkelnummer

contact

Kontaktdaten von Wirtschaftspartnern.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	int(11)	Nein	PK	Fortlaufende Nummer
last_name	varchar(255)	Nein		Nachname
first_name	varchar(255)	Ja		Vorname
company	varchar(255)	Nein		Firmenname
street1	varchar(255)	Ja		Straße (Zeile 1)
street2	varchar(255)	Ja		Straße (Zeile 2)
postal_code	int(5)	Ja		Postleitzahl
city	varchar(255)	Ja		Stadt
state	varchar(255)	Ja		Bundesland
country	varchar(255)	Ja		Land
phone1	varchar(255)	Ja		Telefonnummer 1
phone2	varchar(255)	Ja		Telefonnummer 2
fax	varchar(255)	Ja		Faxnummer
email	varchar(64)	Ja		E-Mail Adresse
website	varchar(64)	Ja		Adresse der Webseite

diagnosis

Liste der Diagnosen und Krankheiten.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	tinyint(4)	Nein	PK	Fortlaufende Nummer
name	varchar(255)	Nein		Name der Krankheit/Diagnose

entity_1

Daten empfangener ADIS/ADED Entitäten vom Typ entity_1 (environment_values).

Feld	Typ	Null	Besonderheit	Kommentar
id	bigint(20)	Nein	PK	Fortlaufende Nummer
farm	varchar(20)	Nein		Betriebsnummer (lokal = 0)
stable	varchar(20)	Nein		Stallnummer
compartment	varchar(20)	Nein		Abteilnummer
pen	varchar(20)	Nein		Buchtnummer
timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Messzeitpunkt
sender	varchar(24)	Nein		Gerät das Wert geschickt hat

Feld	Typ	Null	Besonderheit	Kommentar
coordinate_x	float	Ja		X Koordinate
coordinate_y	float	Ja		Y Koordinate
coordinate_z	float	Ja		Z Koordinate
temperature	float	Ja		Temperaturwert
humidity	float	Ja		Feuchtigkeitswert
co2	float	Ja		CO2 Wert
nh3	float	Ja		NH3 Wert
ch4	float	Ja		CH4 Wert
n2o	float	Ja		N2O Wert
unterdruck	float	Ja		Wert des Unterdrucks
helligkeit	float	Ja		Helligkeitswert
air_speed	float	Ja		Luftgeschwindigkeit

entity_2
Daten empfangener ADIS/ADED Entitäten vom Typ entity_2 (meter_values).

Feld	Typ	Null	Besonderheit	Kommentar
id	bigint(20)	Nein	PK	Fortlaufende Nummer
farm	varchar(20)	Nein		Betriebsnummer (lokal = 0)
stable	varchar(20)	Nein		Stallnummer
compartment	varchar(20)	Nein		Abteilnummer
pen	varchar(20)	Nein		Buchtnummer
timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Messzeitpunkt
sender	varchar(24)	Nein		Gerät das Wert geschickt hat
coordinate_x	float	Ja		X Koordinate
coordinate_y	float	Ja		Y Koordinate
coordinate_z	float	Ja		Z Koordinate
electric_meter	float	Ja		Verbrauchswert Strommesser
water_meter	float	Ja		Verbrauchswert Wassermesser
heat_meter	float	Ja		Verbrauchswert Wärmemesser
gas_meter	float	Ja		Verbrauchswert Gasmesser

entity_101000
Daten empfangener ADIS/ADED Entitäten vom Typ entity_101000 (Klima_Luftdaten).

Feld	Typ	Null	Besonderheit	Kommentar
id	bigint(20)	Nein	PK	Fortlaufende Nummer
farm	varchar(20)	Nein		Betriebsnummer (lokal = 0)
stable	varchar(20)	Nein		Stallnummer
compartment	varchar(20)	Nein		Abteilnummer
pen	varchar(20)	Nein		Buchtnummer
timestamp	timestamp	Nein	DV: CURRENT_	Messzeitpunkt

Feld	Typ	Null	Besonderheit	Kommentar
			TIMESTAMP	
sender	varchar(24)	Nein		Gerät das Wert geschickt hat
temperature	float	Nein		Temperaturwert
humidity	float	Ja		Feuchtigkeitswert
co2	float	Ja		CO2 Wert
nh3	float	Ja		NH3 Wert

entity_101001
Daten empfangener ADIS/ADED Entitäten vom Typ entity_101001 (Klima_Aussen).

Feld	Typ	Null	Besonderheit	Kommentar
id	bigint(20)	Nein	PK	Fortlaufende Nummer
farm	varchar(20)	Nein		Betriebsnummer (lokal = 0)
stable	varchar(20)	Nein		Stallnummer
compartment	varchar(20)	Nein		Abteilnummer
pen	varchar(20)	Nein		Buchtnummer
timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Messzeitpunkt
sender	varchar(24)	Nein		Gerät das Wert geschickt hat
temperature	float	Nein		Temperaturwert
humidity	float	Ja		Feuchtigkeitswert
wind	float	Ja		Windstärke
windrichtung	float	Ja		Windrichtung
luftdruck	float	Ja		Luftdruck
helligkeit	float	Ja		Helligkeit
regen	float	Ja		Regenstärke
enthalpie	float	Ja		Enthalpie

entity_101005
Daten empfangener ADIS/ADED Entitäten vom Typ entity_101005 (Klima_Tierdaten).

Feld	Typ	Null	Besonderheit	Kommentar
id	bigint(20)	Nein	PK	Fortlaufende Nummer
farm	varchar(20)	Nein		Betriebsnummer (lokal = 0)
stable	varchar(20)	Nein		Stallnummer
compartment	varchar(20)	Nein		Abteilnummer
pen	varchar(20)	Nein		Buchtnummer
timestamp	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Messzeitpunkt
sender	varchar(24)	Nein		Gerät das Wert geschickt hat
tierart	float	Ja	FK	Tierart (nicht verwendet)
age	float	Ja		Alter in Tagen
weight	float	Ja		Gewicht
s_anz_g2	float	Ja		

Feld	Typ	Null	Besonderheit	Kommentar
wasser	float	Ja		Wasserverbrauch

entity_610011
Daten empfangener ADIS/ADED Entitäten vom Typ entity_610011 (WIEGEN).

Feld	Typ	Null	Besonderheit	Kommentar
tier_id	bigint(15)	Nein	PK, FK	Elektronische ID aus Tabelle animal
WIEGEDATUM	date	Nein		Wiegedatum
WIEGEZEIT	time	Ja		Wiegezeit
GEWICHT	float	Nein		Gewicht in kg
GEWICHTSART	tinyint(4)	Ja		
GEWICHTSTYP	tinyint(4)	Ja		
sender	varchar(24)	Nein		Gerät das Wert geschickt hat

feed_consumption
Futterverbräuche.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	bigint(20)	Nein	PK	Fortlaufende Nummer
penNumber	tinyint(4)	Nein		Buchtnummer
numberOfAnimals	tinyint(4)	Nein		Tieranzahl
valveConsumption	int(11)	Nein		Futtermenge in g
date	date	Nein		Datum

google_chart_url
Google Chart URLs der durch den Info Mailer erzeugten Grafiken.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	bigint(20)	Nein	PK	Fortlaufende Nummer
f_location_id	tinyint(4)	Nein	FK	ID aus Tabelle location
f_sensor_id	int(11)	Nein	FK	ID aus Tabelle sensor
start_time	timestamp	Nein		Startzeitpunkt der Darstellung
end_time	timestamp	Nein		Endzeitpunkt der Darstellung
url	text	Nein		URL des Google Charts

leaving_reason
Gründe für den Abgang von Masttieren.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	tinyint(4)	Nein	PK	Fortlaufende Nummer
description	varchar(64)	Nein		Abgangsgrund

location
Mögliche Farm/Stall/Abteil/Bucht Kombinationen.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	tinyint(4)	Nein	PK	Fortlaufende Nummer
stable_number	varchar(15)	Nein		Nummer des Stalls

Feld	Typ	Null	Besonderheit	Kommentar
compartment_number	varchar(15)	Nein		Nummer des Abteils
pen_number	varchar(15)	Nein		Nummer der Bucht
farm_number	varchar(15)	Nein		Nummer der Farm; 0 bei lokal
description	varchar(128)	Nein		Beschreibender Text

Da der ISOagriNET Adapter für die Lüftungssteuerung der Firma Möller bis zum Ende der Implementierung aller Dienste nur in der Lage war, feste Werte (farmnumber = Testkunde, stable = 1) zu senden, werden die Attribute stable_number und farm_number für die View Erzeugung nicht berücksichtigt. Für die Farming Cell ist dies solange unproblematisch, wie sie nicht erweitert oder auf andere Betriebe ausgedehnt wird. Mit der neuen Firmware Version des Adapters vom September 2009 ist die genannte Einschränkung beseitigt worden. Die Datenbankseitige Anpassung ist nicht mehr erfolgt.

Einträge in der Tabelle location.

farm_number	stable_number	compartment_number	pen_number	description
0	3	1	1	Bucht 1
0	3	1	2	Bucht 2
0	3	2	3	Bucht 3
0	3	2	4	Bucht 4
0	3	1	0	Abteil 1
0	3	2	0	Abteil 2
0	0	0	0	Außen/Vorraum

measuring_point

Alle ehemaligen und aktuellen Messpunkte.

Feld	Typ	Null	Besonderheit	Kommentar
id	int(11)	Nein	PK	Fortlaufende Nummer
coordinate_x	float	Ja		X Koordinate
coordinate_y	float	Ja		Y Koordinate
coordinate_z	float	Ja		Z Koordinate
valid_from	timestamp	Nein	DV: CURRENT_ TIMESTAMP	Zeitpunkt ab dem der Messpunkt aktiv ist
valid_till	timestamp	Ja		Zeitpunkt bis zu dem der Messpunkt aktiv ist
description	varchar(255)	Nein		Beschreibender Text

medical_treatment

Durchgeführte Behandlungen.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	bigint(20)	Nein	PK	Fortlaufende Nummer
f_electronical_anima l_id	bigint(20)	Nein	FK	ID des Tieres aus Tabelle animal

Feld	Typ	Null	Besonderheit	Kommentar
f_medicine_id	tinyint(4)	Ja	FK	ID der Medizin aus Tabelle medicine
AuA_number	bigint(20)	Ja		AuA Nummer des Medikaments
quantity	tinyint(4)	Ja		Verabreichte Menge in ml
f_diagnosis	tinyint(4)	Nein	FK	ID der Krankheit/Diagnose aus Tabelle diagnosis
comment	varchar(255)	Nein		Diagnose / Kommentar zur Behandlung
f_user_id	tinyint(4)	Nein	FK	ID des Eintragenden aus Tabelle user
timestamp	timestamp	Nein	DV: CURRENT_TIMESTAMP	Behandlungszeitpunkt

medicine
Medikamente.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	int(11)	Nein	PK	Fortlaufende Nummer
description	varchar(255)	Nein		Name des Medikaments
waiting_time	tinyint(4)	Nein		Wartezeit in Tagen

race
Vorkommende Schweinerassen.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	tinyint(4)	Nein	PK	Fortlaufende Nummer
name	varchar(32)	Nein		Rassename

sensors
In der Farming Cell zulässige Sensortypen.

Feld	Typ	Null	Besonderheit	Kommentar
id	tinyint(4)	Nein	PK	Fortlaufende Nummer
name_intern	varchar(64)	Nein		Sensorbezeichnung gemäß ADIS/ADED
name_extern	varchar(64)	Nein		Im Webfrontend und in Reports Anzuzeigende Bezeichnung
unit	varchar(64)	Nein		Messeinheit
control_zone_min	float	Ja		Untergrenze des Kontrollbereichs
control_zone_max	float	Ja		Obergrenze des Kontrollbereichs

temp_animal
Alle potentiellen Masttiere aus der Software Supersau.

Feld	Typ	Null	Besonderheit	Kommentar
mother_no	varchar(10)	Nein		Nummer der Mutter
piglet_no	tinyint(4)	Nein		Nummer des Ferkels
f_father_id	tinyint(4)	Nein	FK	ID aus Tabelle boar
date_of_birth	date	Nein		Geburtsdatum
sex	tinyint(4)	Nein		Geschlecht; 1 = männlich 2 =

Feld	Typ	Null	Besonderheit	Kommentar
				weiblich
f_race_id	tinyint(4)	Nein	FK	ID aus Tabelle race

user
Webinterface Benutzer.

Feld	Typ	Null	Besonderheit	Kommentar
p_id	tinyint(4)	Nein	PK	Fortlaufende Nummer
login_name	varchar(20)	Nein		Anmeldename
password	varchar(64)	Nein		Anmeldepasswort als MD5 Hash
admin	tinyint(1)	Nein		1 wenn Admin, sonst 0
forename	varchar(64)	Nein		Vorname
surname	varchar(64)	Nein		Nachname

Attributdefinitionen
Nachfolgend sind alle in den Tabellen der Datenbank vorkommenden Attribute in alphabetischer Reihenfolge aufgelistet und kurz erläutert. Die Tatsache, dass sowohl englische als auch deutsche Attributnamen vorkommen, sowie ein Teil groß und der andere Teil kleingeschrieben ist, liegt in der jeweiligen Herkunft begründet. Generell gilt: kleingeschriebene englische Attributnamen wurden im Rahmen des Datenbankdesigns selbst vergeben; deutsche und / oder großgeschriebene Namen sind aus dem ISOagriNET Data Dictionary des LKV NRW übernommen.

admin
Tabellen: user
Indikator ob Webfrontend Benutzer Administratorenrechte hat.

age
Tabellen: entity_101005
Tierdurchschnittsalter in Tagen.

air_speed
Tabellen: entity_1
Ein Messwert.

arrived_timestamp
Tabellen: animal, animal_to_location
Datum und Uhrzeit wann das Tier eingestallt wurde (Tabelle animal) oder wann es einer Lokation zugeordnet wurde (Tabelle animal_to_location).

AuA_number
Tabellen: medical_treatment
Die Anwendungs- und Abgabebeleg (AuA) Nummer der Medizinflasche.

ch4
Tabellen: entity_1
Ein Messwert.

city
Tabellen: contact
Die Stadt eines Kontaktes.

co2
Tabellen: entity_1, entity_101000
Ein Messwert.

comment
Tabellen: medical_treatment
Die Diagnose oder ein Kommentar zur Behandlung.

company
Tabellen: contact
Ein Firmenname des Kontaktes.

compartment
Tabellen: entity_1, entity_2, entity_101000, entity_101001, entity_101005
Eine alphanumerische Kennung eines Abteils.

compartment_number
Tabellen: location
Eine alphanumerische Kennung eines Abteils.

control_zone_max, control_zone_min
Tabellen: sensors
Ober- und Untergrenze des Kontrollbereichs eines Sensors die in der täglichen Info-Mail berücksichtigt werden.

coordinate_x, coordinate_y, coordinate_z
Tabellen: measuring_point, entity_1, entity_2
Die Koordinaten eines Messpunktes.

country
Tabellen: contact
Das Land eines Kontaktes.

date
Tabellen: feed_consumption
Das Datum im Format YYYY-MM-DD für das der Verbrauch gilt.

date_of_birth
Tabellen: Tabellen: animal , temp_animal
Das Geburtsdatum eines Tieres.

description
Tabellen: medicine, measuring_point, location, leaving_reason
Name des Medikaments (Tabelle medicine), Beschreibung des Messpunktes (Tabelle measuring_point), Beschreibung einer Lokation (Tabelle location), Abgangsgrund (Tabelle leaving_reason)

electric_meter
Tabellen: entity_2
Ein Messwert.

email
Tabellen: contact
Ein Emailadresse des Kontaktes.

end_time
Tabellen: google_chart_url
Zeitliches Ende der Darstellung.

enthalpie
Tabellen: entity_101001
Ein Messwert.

father_id
Tabellen: boar (Feld ID)
Schlüssel in der Software Supersau, welcher in die Datenbank der Farming Cell übernommen wird (Feld ID der Tabelle boar)). Diese Nummer identifiziert einen Eber eindeutig.

f_diagnosis, f_e_animal_id, f_electronical_animal_id, f_father_id, f_leaving_contact_id, f_leaving_reason_id, f_location_id, f_medicine_id, f_race_id, f_sensor_id, f_user_id
Die Fremdschlüssel sind der Nomenklatur f_[Herkunftstabellenname]_id entsprechend zusammengesetzt. Das vorangestellte f bedeutet foreign (engl. fremd).

farm
Tabellen: entity_1, entity_2, entity_101000, entity_101001, entity_101005
Eine alphanumerische Kennung eines Betriebes (lokaler Betrieb = 0).

farm_number
Tabellen: location
Eine alphanumerische Kennung eines Betriebes (lokaler Betrieb = 0).

fax
Tabellen: contact
Die Faxnummer eines Kontaktes.

first_name
Tabellen: contact
Der Vorname des Kontaktes.

forename
Tabellen: user
Vorname eines Webfrontend Benutzers.

gas_meter
Tabellen: entity_2
Ein Messwert.

GEWICHT
Tabellen: entity_610011
Das ermittelte Gewicht.

GEWICHTSART
Tabellen: entity_610011
Tot- oder Lebendgewicht.

GEWICHTSTYP
Tabellen: entity_610011
Einzeltier- oder Gruppengewicht.

heat_meter
Tabellen: entity_2
Ein Messwert.

helligkeit
Tabellen: entity_1, entity_101001
Ein Messwert.

humidity
Tabellen: entity_1, entity_101000, entità_101001
Ein Messwert.

id
Tabellen: sensors, measuring_point, entity_1, entity_2, entity_101000, entity_101001, entity_101005
Fortlaufende Nummer, die als Primärschlüssel für die Einträge der jeweiligen Tabelle dient.

last_name
Tabellen: contact
Der Nachname des Kontaktes.

left_timestamp
Tabellen: animal, animal_to_location
Datum und Uhrzeit wann das Tier ausgestallt wurde um den Betrieb zu verlassen (Tabelle animal) oder wann es einer Lokation zugeordnet wurde (Tabelle animal_to_location).

login_name
Tabellen: user
Name eines Webfrontend Benutzers.

luftdruck
Tabellen: entity_101001
Ein Messwert.

mother_no
Tabellen: animal , temp_animal, boar
Schlüssel in der Software Supersau, welcher in die Datenbank der Farming Cell übernommen wird. Diese alphanumerische Zeichenfolge identifiziert die Muttersau eindeutig.

n2o
Tabellen: entity_1
Ein Messwert.

name
Tabellen: race, diagnosis, boar
Name der Rasse (Tabelle race), Name der Krankheit, der Diagnose (Tabelle diagnosis) oder des Ebers (Tabelle boar).

name_extern
Tabellen: sensors
Die im Webfrontend anzuzeigende Bezeichnung eines Sensors oder Verbrauchsmessers.

name_intern
Tabellen: sensors
Bezeichnung eines Sensors oder Verbrauchsmessers entsprechend ADIS/ADED. Es existieren deutsche und englische Bezeichnungen, wobei die im Projektrahmen hinzugefügten Bezeichnungen grundsätzlich englischsprachig sind.

nh3
Tabellen: entity_1, entity_101000
Ein Messwert.

numberOfAnimals
Tabellen: feed_consumption
Anzahl der Tiere auf die sich der Futterverbrauch bezieht.

p_electronical_id
Tabellen: animal
Die elektronische Nummer einer RFID Ohrmarke.

p_id
Tabellen: user, race, medicine, medical_treatment, location, leaving_reason, google_chart_url, feed_consumption, diagnosis, boar, contact
Fortlaufende Nummer, die als Primärschlüssel für die Einträge der jeweiligen Tabelle dient.

password
Tabellen: user
Das MD5 verschlüsselte Passwort eines Webfrontend Benutzers.

pen
Tabellen: entity_1, entity_2, entity_101000, entity_101001, entity_101005
Eine alphanumerische Kennung einer Bucht.

pen_number
Tabellen: location
Eine alphanumerische Kennung einer Bucht.

penNumber
Tabellen: feed_consumption
Nummer der Bucht auf die sich der Futterverbrauch bezieht.

phone1, phone2
Tabellen: contact
Die Telefonnummern eines Kontaktes.

piglet_no
Tabellen: animal, temp_animal, boar
Die Nummer eines Ferkels. Die Nummern werden für die Ferkel jeder Sau wurfübergreifend und fortlaufend vergeben.

postal_code
Tabellen: contact
Die Postleitzahl.

quantity
Tabellen: medical_treatment
Die verabreichte Menge eines Medikaments.

regen
Tabellen: entity_101001
Ein Messwert.

s_anz_g2
Tabellen: entity_101005
Anzahl Tiere in einer Lokation.

sender
Tabellen: entity_1, entity_2, entity_101000, entity_101001, entity_101005, entity_610011
Name oder Identifier des Gerätes oder der Software die der Ursprung eines Wertes ist.

sex
Tabellen: animal, temp_animal
Das Geschlecht eines Masttieres.

stable
Tabellen: entity_1, entity_2, entity_101000, entity_101001, entity_101005
Eine alphanumerische Kennung eines Stalls.

stable_number
Tabellen: location
Eine alphanumerische Kennung eines Stalls.

start_time
Tabellen: google_chart_url
Zeitlicher Begin der Darstellung.

state
Tabellen: contact
Das Bundesland eines Kontaktes.

street1, street2
Tabellen: contact
Die Straßennamen eines Kontaktes.

surname
Tabellen: user
Nachname eines Webfrontend Benutzers.

temperature
Tabellen: entity_1, entity_101000, entità_101001
Ein Messwert.

tier_id
Tabellen: entity_610011
Die elektronische Nummer der RFID Ohrmarke eines Tieres.

tierart
Tabellen: entity_101005
Die Tierart entsprechend ADIS/ADED Codeset 1101.

timestamp
Tabellen: medical_treatment, entity_1, entity_2, entity_101000, entity_101001, entity_101005
Ein Behandlungszeitpunkt (Tabelle medical_treatment), Messzeitpunkt (andere Tabellen).

unit
Tabellen: sensors
Die Einheit der zu dem Sensor oder Verbrauchsmesser gehörenden Messwerte.

unterdruck
Tabellen: entity_1
Ein Messwert.

url
Tabellen: google_chart_url
Die URL einer Webseite.

valid_from, valid_till
Tabellen: measuring_point
Start- und Enddatum sowie Uhrzeit der Gültigkeit eines Messpunktes.

valveConsumption
Tabellen: feed_consumption
Eine Futtermenge, die durch das Ventil (eng. valve) geflossen ist. Ein Ventil 1-4 entspricht im konkreten Fall der Bucht 1-4.

visual_id
Tabellen: animal
Die tierindividuelle Nummer auf der Ohrmarke.

waiting_time
Tabellen: medicine, animal
Die Anzahl Tage die ein Tier nach einer Behandlung zu sperren ist (Tabelle medicine) oder das Datum bis zum dem ein Tier aufgrund einer Medikamentierung für den Weiterverkauf gesperrt ist (Tabelle animal).

wasser
Tabellen: entity_101005
Wasserverbrauch in einer Lokation an einem Tag.

water_meter
Tabellen: entity_2
Ein Messwert.

website
Tabellen: contact
Die Webseite eines Kontaktes.

weight
Tabellen: entity_101005
Tierdurchschnittsgewicht pro Tier in Kilogramm.

WIEGEDATUM
Tabellen: entity_610011
Datum eines Wiegeereignisses.

WIEGEZEIT
Tabellen: entity_610011
Uhrzeit eines Wiegeereignisses.

wind
Tabellen: entity_101001
Ein Messwert.

windrichtung
Tabellen: entity_101001
Ein Messwert.

Create Statements der Datenbank Views
Da die Herkunft der Daten in den Views der Farming Cell Datenbank nicht ersichtlich ist, sind alle für die View Erzeugung notwendigen SQL Statements in diesem Abschnitt hinterlegt.

view_temperature
```
CREATE VIEW view_temperature AS
SELECT table1.temperature AS value, table1.timestamp, table1.sender, table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL, table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL, table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL, table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND (IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.temperature IS NOT NULL
UNION ALL
SELECT table1.temperature AS value, table1.timestamp, table1.sender, table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101000 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND (IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.temperature IS NOT NULL
UNION ALL
SELECT table1.temperature AS value, table1.timestamp, table1.sender, table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND (IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.temperature IS NOT NULL
```

view_humidity
```
CREATE VIEW view_humidity AS
SELECT table1.humidity AS value, table1.timestamp, table1.sender, table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL, table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL, table1.coordinate_y = table3.coordinate_y)
```

```
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.humidity IS NOT NULL
UNION ALL
SELECT table1.humidity AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101000 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.humidity IS NOT NULL
UNION ALL
SELECT table1.humidity AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.humidity IS NOT NULL
```

view_co2

```
CREATE VIEW view_co2 AS
SELECT table1.co2 AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.co2 IS NOT NULL
UNION ALL
SELECT table1.co2 AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_101000 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), isnull() ,table3.valid_till) )
AND table1.co2 IS NOT NULL
```

view_nh3

```
CREATE VIEW view_nh3 AS
SELECT table1.nh3 AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND    IF(isnull(table1.coordinate_x),    table3.coordinate_x    IS    NULL,
table1.coordinate_x = table3.coordinate_x)
AND    IF(isnull(table1.coordinate_y),    table3.coordinate_y    IS    NULL,
table1.coordinate_y = table3.coordinate_y)
AND    IF(isnull(table1.coordinate_z),    table3.coordinate_z    IS    NULL,
table1.coordinate_z = table3.coordinate_z)
AND       table1.timestamp       BETWEEN       table3.valid_from       AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.nh3 IS NOT NULL
UNION ALL
SELECT table1.nh3 AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_101000 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND       table1.timestamp       BETWEEN       table3.valid_from       AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.nh3 IS NOT NULL
```

view_ch4

```
CREATE VIEW view_ch4 AS
SELECT table1.ch4 AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND    IF(isnull(table1.coordinate_x),    table3.coordinate_x    IS    NULL,
table1.coordinate_x = table3.coordinate_x)
AND    IF(isnull(table1.coordinate_y),    table3.coordinate_y    IS    NULL,
table1.coordinate_y = table3.coordinate_y)
AND    IF(isnull(table1.coordinate_z),    table3.coordinate_z    IS    NULL,
table1.coordinate_z = table3.coordinate_z)
AND       table1.timestamp       BETWEEN       table3.valid_from       AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.ch4 IS NOT NULL
```

view_n2o

```
CREATE                          VIEW                          view_n2o
(value,timestamp,sender,f_location_id,f_measuring_point_id) AS
SELECT table1.n2o AS value, table1.timestamp, table1.sender, table2.p_id AS
f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND    IF(isnull(table1.coordinate_x),    table3.coordinate_x    IS    NULL,
table1.coordinate_x = table3.coordinate_x)
AND    IF(isnull(table1.coordinate_y),    table3.coordinate_y    IS    NULL,
table1.coordinate_y = table3.coordinate_y)
```

```
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.n2o IS NOT NULL
```

view_helligkeit

```
CREATE VIEW view_helligkeit AS
SELECT table1.helligkeit AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.helligkeit IS NOT NULL
UNION ALL
SELECT table1.helligkeit AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.helligkeit IS NOT NULL
```

view_unterdruck

```
CREATE VIEW view_unterdruck AS
SELECT table1.unterdruck AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.unterdruck IS NOT NULL
```

view_air_speed

```
CREATE VIEW view_air_speed AS
SELECT table1.air_speed AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_1 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
```

```
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.air_speed IS NOT NULL
```

view_wind

```
CREATE VIEW view_wind AS
SELECT table1.wind AS value, table1.timestamp, table1.sender, table2.p_id
AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.wind IS NOT NULL
```

view_windrichtung

```
CREATE VIEW view_windrichtung AS
SELECT table1.windrichtung AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.windrichtung IS NOT NULL
```

view_luftdruck

```
CREATE VIEW view_luftdruck AS
SELECT table1.luftdruck AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.luftdruck IS NOT NULL
```

view_regen

```
CREATE VIEW view_regen AS
SELECT table1.regen AS value, table1.timestamp, table1.sender, table2.p_id
AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
```

```
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.regen IS NOT NULL
```

view_enthalpie

```
CREATE VIEW view_enthalpie AS
SELECT table1.enthalpie AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_101001 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table3.coordinate_x IS NULL
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.enthalpie IS NOT NULL
```

view_electric_meter

```
CREATE VIEW view_electric_meter AS
SELECT table1.electric_meter AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_2 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.electric_meter IS NOT NULL
```

view_water_meter

```
CREATE VIEW view_water_meter AS
SELECT table1.water_meter AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_2 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.water_meter IS NOT NULL
```

view_heat_meter

```
CREATE VIEW view_heat_meter AS
SELECT table1.heat_meter AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_2 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.heat_meter IS NOT NULL
```

view_gas_meter

```
CREATE VIEW view_gas_meter AS
SELECT table1.gas_meter AS value, table1.timestamp, table1.sender,
table2.p_id AS f_location_id, table3.id AS f_measuring_point_id
FROM entity_2 table1, location table2, measuring_point table3
WHERE
table1.compartment = table2.compartment_number
AND table1.pen = table2.pen_number
AND IF(isnull(table1.coordinate_x), table3.coordinate_x IS NULL,
table1.coordinate_x = table3.coordinate_x)
AND IF(isnull(table1.coordinate_y), table3.coordinate_y IS NULL,
table1.coordinate_y = table3.coordinate_y)
AND IF(isnull(table1.coordinate_z), table3.coordinate_z IS NULL,
table1.coordinate_z = table3.coordinate_z)
AND table1.timestamp BETWEEN table3.valid_from AND
(IF(isnull(table3.valid_till), NOW() ,table3.valid_till) )
AND table1.gas_meter IS NOT NULL
```

view_sensor_to_location

```
CREATE VIEW view_sensor_to_location AS
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'temperature') as f_sensor_id , "Temperatur" as sensor FROM
`view_temperature`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'humidity') as f_sensor_id, "Feuchtigkeit" as sensor FROM
`view_humidity`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'co2') as f_sensor_id, "CO2" as sensor FROM `view_co2`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'nh3') as f_sensor_id, "NH3" as sensor FROM `view_nh3`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'ch4') as f_sensor_id, "CH4" as sensor FROM `view_ch4`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'n2o') as f_sensor_id, "N2O" as sensor FROM `view_n2o`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'helligkeit') as f_sensor_id, "Helligkeit" as sensor FROM
`view_helligkeit`
UNION ALL
```

```
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'unterdruck') as f_sensor_id, "Unterdruck" as sensor FROM
`view_unterdruck`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'air_speed') as f_sensor_id, "Lüftungsgeschwindigkeit" as
sensor FROM `view_air_speed`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'wind') as f_sensor_id, "Windstärke" as sensor FROM
`view_wind`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'windrichtung') as f_sensor_id, "Windrichtung" as sensor
FROM `view_windrichtung`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'luftdruck') as f_sensor_id, "Luftdruck" as sensor FROM
`view_luftdruck`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'regen') as f_sensor_id, "Regen" as sensor FROM
`view_regen`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'enthalpie') as f_sensor_id, "Enthalpie" as sensor FROM
`view_enthalpie`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'electric_meter') as f_sensor_id, "Stromzähler" as sensor
FROM `view_electric_meter`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'water_meter') as f_sensor_id, "Wasserzähler" as sensor
FROM `view_water_meter`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'heat_meter') as f_sensor_id, "Wärmezähler" as sensor FROM
`view_heat_meter`
UNION ALL
SELECT DISTINCT `f_location_id`, (SELECT `id` FROM `sensors` WHERE
`name_intern` = 'gas_meter') as f_sensor_id, "Gaszähler" as sensor FROM
`view_gas_meter`
```

view avg_temperature

```
CREATE VIEW view_avg_temperature AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_temperature`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_humidity

```
CREATE VIEW view_avg_humidity AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_humidity`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_co2

```
CREATE VIEW view_avg_co2 AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_co2`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_nh3

```
CREATE VIEW view_avg_nh3 AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_nh3`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_ch4

```
CREATE VIEW view_avg_ch4 AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_ch4`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_n2o

```
CREATE VIEW view_avg_n2o AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_n2o`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_helligkeit

```
CREATE VIEW view_avg_helligkeit AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_helligkeit`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_unterdruck

```
CREATE VIEW view_avg_unterdruck AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS value, COUNT(value) AS no_of_values
FROM `view_unterdruck`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id, f_measuring_point_id
```

view avg_air_speed

```
CREATE VIEW view_avg_air_speed AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS no_of_values
FROM `view_air_speed`
```

```
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_wind

```
CREATE VIEW view_avg_wind AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_wind`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_windrichtung

```
CREATE VIEW view_avg_windrichtung AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_windrichtung`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_luftdruck

```
CREATE VIEW view_avg_luftdruck AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_luftdruck`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_regen

```
CREATE VIEW view_avg_regen AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_regen`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_enthalpie

```
CREATE VIEW view_avg_enthalpie AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, avg(value) AS value, COUNT(value) AS
no_of_values
FROM `view_enthalpie`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_electric_meter

```
CREATE VIEW view_avg_electric_meter AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, SUM( value ) AS value, COUNT( value )
AS no_of_values
FROM `view_electric_meter`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_water_meter

```
CREATE VIEW view_avg_water_meter AS
```

```
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, SUM( value ) AS value, COUNT( value )
AS no_of_values
FROM `view_water_meter`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_heat_meter

```
CREATE VIEW view_avg_heat_meter AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, SUM( value ) AS value, COUNT( value )
AS no_of_values
FROM `view_heat_meter`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view avg_gas_meter

```
CREATE VIEW view_avg_gas_meter AS
SELECT `f_location_id` , `f_measuring_point_id` , DATE_FORMAT( timestamp,
'%Y-%m-%d %H:00:00' ) AS timestamp, SUM( value ) AS value, COUNT( value )
AS no_of_values
FROM `view_gas_meter`
GROUP BY DATE_FORMAT( timestamp, '%Y-%m-%d %H:00:00' ) , f_location_id,
f_measuring_point_id
```

view_pig_total_group_weight

```
CREATE VIEW view_pig_total_group_weight AS
SELECT CAST(SUM( table1.GEWICHT) AS DECIMAL (7,2)) AS value, CAST( CONCAT(
table1.WIEGEDATUM, ' 00:00:00' ) AS DATETIME ) AS timestamp, table1.sender
AS sender, table2.f_location_id, "19" AS f_measuring_point_id
FROM entity_610011 table1, animal_to_location table2
WHERE table1.tier_id = table2.f_e_animal_id
AND CONCAT( table1.WIEGEDATUM, ' 23:59:58'' )
BETWEEN table2.arrived_timestamp
AND (
IF( isnull( table2.left_timestamp ) , CONCAT(CURDATE(), ' 23:59:59') ,
table2.left_timestamp )
)
GROUP BY table1.WIEGEDATUM, table2.f_location_id
```

view_pig_avg_pig_weight

```
CREATE VIEW view_pig_avg_pig_weight AS
SELECT CAST(SUM( table1.GEWICHT)/COUNT(table1.GEWICHT) AS DECIMAL(7,2)) AS
value, CAST( CONCAT( table1.WIEGEDATUM, ' 00:00:00' ) AS DATETIME ) AS
timestamp, table1.sender AS sender, table2.f_location_id, "19" AS
f_measuring_point_id
FROM entity_610011 table1, animal_to_location table2
WHERE table1.tier_id = table2.f_e_animal_id
AND CONCAT( table1.WIEGEDATUM, ' 23:59:58' )
BETWEEN table2.arrived_timestamp
AND (
IF( isnull( table2.left_timestamp ) , CONCAT(CURDATE(), ' 23:59:59') ,
table2.left_timestamp )
)
GROUP BY table1.WIEGEDATUM, table2.f_location_id
```

view_pig_total_group_feed_consumption

```
CREATE VIEW view_pig_total_group_feed_consumption AS
SELECT CAST( table1.valveConsumption AS DECIMAL( 7, 2 ) ) AS value, CAST(
CONCAT( table1.date, ' 00:00:00' ) AS DATETIME ) AS timestamp, "Schauer" AS
sender, table2.p_id AS f_location_id, "19" AS f_measuring_point_id
```

```
FROM feed_consumption table1, location table2
WHERE table1.penNumber = table2.pen_number
GROUP BY table1.date, table2.p_id
```

view_pig_avg_pig_feed_consumption

```
CREATE VIEW view_pig_avg_pig_feed_consumption AS
SELECT CAST( table1.valveConsumption/table1.numberOfAnimals AS DECIMAL( 7,
2 ) ) AS value, CAST( CONCAT( table1.date, ' 00:00:00' ) AS DATETIME ) AS
timestamp, "Trutest" AS sender, table2.p_id AS f_location_id, "19" AS
f_measuring_point_id
FROM feed_consumption table1, location table2
WHERE table1.penNumber = table2.pen_number
GROUP BY table1.date, table2.p_id
```

view_pig_daily_gain

```
CREATE VIEW view_pig_daily_gain AS
SELECT DISTINCT (
IFNULL( (
(
value - (
SELECT value
FROM `view_pig_avg_pig_weight` table3
WHERE table3.value < table2.value
AND table3.f_location_id = table2.f_location_id
AND table3.f_measuring_point_id = table2.f_measuring_point_id
ORDER BY value DESC
LIMIT 0 , 1
) ) / DATEDIFF( timestamp, (
SELECT timestamp
FROM `view_pig_avg_pig_weight` table1
WHERE table1.timestamp < table2.timestamp
AND table1.f_location_id = table2.f_location_id
AND table1.f_measuring_point_id = table2.f_measuring_point_id
ORDER BY timestamp DESC
LIMIT 0 , 1
) )
), 0
)
) AS value, table2.timestamp, `f_location_id` , `f_measuring_point_id`
FROM `view_pig_avg_pig_weight` table2
```

view_last_pig_weight_pen1

```
CREATE VIEW view_last_pig_weight_pen1 AS
SELECT table1.tier_id, table1.WIEGEDATUM, table1.GEWICHT, table3.visual_id,
table3.mother_no, table3.piglet_no
FROM entity_610011 table1, animal_to_location table2, animal table3
WHERE
(
SELECT `WIEGEDATUM`
FROM `entity_610011`
ORDER BY `entity_610011`.`WIEGEDATUM` DESC
LIMIT 1
) = table1.`WIEGEDATUM`
AND table1.tier_id = table2.f_e_animal_id
AND isnull( table2.left_timestamp )
AND table2.f_location_id =1
AND table1.tier_id = table3.p_electronical_id
```

view_last_pig_weight_pen2

```
CREATE VIEW view_last_pig_weight_pen2 AS
SELECT table1.tier_id, table1.WIEGEDATUM, table1.GEWICHT, table3.visual_id,
table3.mother_no, table3.piglet_no
```

```
FROM entity_610011 table1, animal_to_location table2, animal table3
WHERE
(
SELECT `WIEGEDATUM`
FROM `entity_610011`
ORDER BY `entity_610011`.`WIEGEDATUM` DESC
LIMIT 1
) = table1.`WIEGEDATUM`
AND table1.tier_id = table2.f_e_animal_id
AND isnull( table2.left_timestamp )
AND table2.f_location_id =2
AND table1.tier_id = table3.p_electronical_id
```

view_last_pig_weight_pen3

```
CREATE VIEW view_last_pig_weight_pen3 AS
SELECT table1.tier_id, table1.WIEGEDATUM, table1.GEWICHT, table3.visual_id,
table3.mother_no, table3.piglet_no
FROM entity_610011 table1, animal_to_location table2, animal table3
WHERE
(
SELECT `WIEGEDATUM`
FROM `entity_610011`
ORDER BY `entity_610011`.`WIEGEDATUM` DESC
LIMIT 1
) = table1.`WIEGEDATUM`
AND table1.tier_id = table2.f_e_animal_id
AND isnull( table2.left_timestamp )
AND table2.f_location_id =3
AND table1.tier_id = table3.p_electronical_id
```

view_last_pig_weight_pen4

```
CREATE VIEW view_last_pig_weight_pen4 AS
 SELECT      table1.tier_id,      table1.WIEGEDATUM,      table1.GEWICHT,
table3.visual_id, table3.mother_no, table3.piglet_no
FROM entity_610011 table1, animal_to_location table2, animal table3
WHERE
(
SELECT `WIEGEDATUM`
FROM `entity_610011`
ORDER BY `entity_610011`.`WIEGEDATUM` DESC
LIMIT 1
) = table1.`WIEGEDATUM`
AND table1.tier_id = table2.f_e_animal_id
AND isnull( table2.left_timestamp )
AND table2.f_location_id =4
AND table1.tier_id = table3.p_electronical_id
```

B – Veröffentlichungen, Arbeitsgruppen, Tagungsteilnahmen, Messeauftritte

Veröffentlichungen

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2009): Farming Cell – Ein ISOagriNET Netzwerk für die Schweinehaltung. Landtechnik 4/2009, 64. Jahrgang, S. 254 – 256.

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2009): Entwicklung, Implementierung und Bewertung von zwei ISOagriNET kompatiblen Systemen zur Messwerterfassung in der landwirtschaftlichen Nutztierhaltung. In Tagungsband: 9. Internationale Tagung „Bau, Technik und Umwelt in der landwirtschaftlichen Nutztierhaltung“. Tagung vom 21.-23. September 2009, Humboldt-Universität zu Berlin.

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2009): Development and evaluation of two ISOagriNET compliant systems for measuring environment and consumption data in animal housing systems. In: Program book of JIAC 2009 Conference.

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2008): Vernetzung von Systemkomponenten und Datenflüssen in der Schweinehaltung. In: elektronische Zeitschrift für Agrarinformatik (eZAI), Band 3 (2008).

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2008): Development of a general principle solution for ISOagriNET compliant networking system components in animal husbandry. In: Proceedings of CCTA 2008 Conference.

Herd, D., A. Kuhlmann, D. Martini, M. Kunisch, E. Friedrichs (2008): Technische Möglichkeiten zur Verbesserung der Prozessdokumentation und Rückverfolgbarkeit in der Schweinehaltung. In Tagungsband: Precision Pig Farming – innovative Technologien und Entscheidungsmodelle für die Schweinehaltung. KTBL Tagung vom 30.09. bis 01.10.2008 in Osnabrück, S. 121 – 131, Darmstadt.

Kuhlmann, A., Herd, D., Rößler, B., Gallmann, E., Jungbluth, T. (2008): Hardwarevernetzung und Softwareintegration in der Schweineproduktion. Landtechnik, 4/2008, 63. Jahrgang, S. 234-235.

Herd D., A. Kuhlmann, B. Rößler, E. Gallmann and T. Jungbluth (2008): Farm Networks in Pig Housing Systems. In: Proceeding of the 9th International Conference on Precision Agriculture, July 20-23 2008, Denver, USA.

Herd, D., Kuhlmann, A., Gallmann, E., Rößler, B., Jungbluth, T. (2008): Networks in Livestock Systems. In: Tagungsband zur AgEng 2008 International Conference on Agricultural Engineering, Kreta, 23.-25. Juni, Griechenland.

Arbeitsgruppe ISOagriNET und agroXML[32]

Die Arbeitsgruppe ISOagriNET und agroXML hat die Aufgabe, die Harmonisierung der beiden Standards voranzutreiben. Da die Datenstrukturen der Standards gegenwärtig noch wachsen, ist eine Abstimmung sinnvoll. Ziel ist es, eine leichte Überführbarkeit von Nachrichten eines Standards in eine Nachricht des jeweils anderen Standards zu ermöglichen.
Die Mitarbeit der Universität Hohenheim in der Arbeitsgruppe erfolgte während der gesamten Projektlaufzeit und wird auch über diese hinaus fortgeführt.

Tagungsteilnahmen

1. GIL 2008, Kiel: Vortrag
2. AgEng 2008, Hersonissos, Kreta: Tagungsteilnahme
3. CCTA 2008, Peking, China: Beitrag und Vortrag
4. JIAC 2009, Wageningen, Niederlande: Beitrag und Vortrag
5. BTU 2009, Berlin: Beitrag und Vortrag

Messeauftritte

AgriTechnica 2007

Präsentation einer Demo eines Portals zum Thema Rückverfolgbarkeit und Qualitätssicherung (Stand der Universität Hohenheim).

EuroTier 2008

Die Präsentation der Farming Cell erfolgte auf dem Stand der Universität Hohenheim wie in Abbildung 0.1 illustriert.

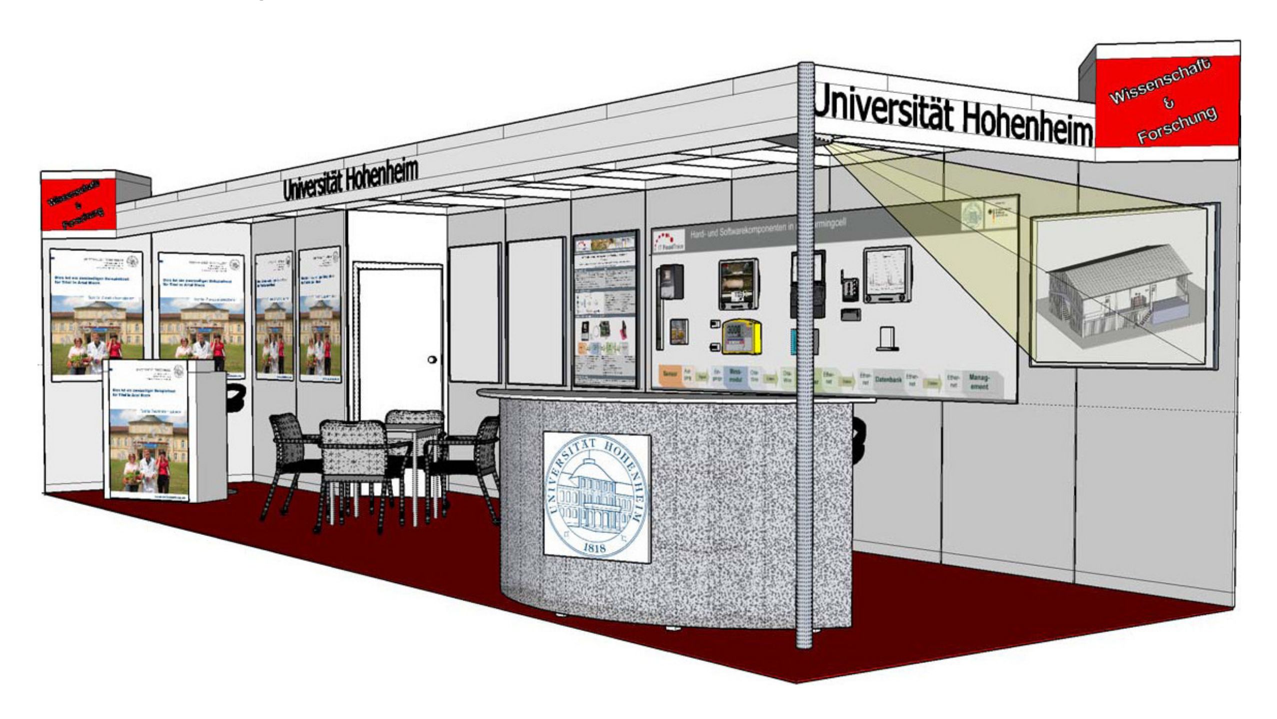


Abbildung 0.2: Stand der Universität Hohenheim auf der EuroTier 2008

[32] Informationen: http://www.ktbl.de/index.php?id=784

CeBIT 2009
Präsentation einer beispielhaften Implementierung einer prozesskettenweiten Architektur für die Sicherstellung der Rückverfolgbarkeit tierischer Erzeugnisse. Der Prototyp wurde durch die Universität Hohenheim, das KTBL, die comundus GmbH und die IBM Deutschland GmbH implementiert und auf den Ständen des Bundesministeriums des Inneren und der IBM Deutschland GmbH präsentiert.
Das Teilprojekt „Informations- und Datengewinnung aus Tierhaltungssystemen" lieferte über ein REST Schnittstelle Betriebs- und Zertifizierungsdaten, welche in einem von der comundus GmbH implementierten Portal angezeigt wurden.